Tampereen yliopisto
Tampere University

TIE-02306

Introduction to Software Engineering

5 credit units

10-projman-ItSE2019-v4

Tampereen yliopisto
Tampere University

Course contents (plan)

1. Course basics, intro
2. Sw Eng in general, overview
3. Requirements
4. Different software systems
5. Basic UML Diagrams (”Class”, Use Case, Navigation)
6. Life Cycle models
7. UML diagrams, in more detail
8. Quality and Testing
9. Project work
10. **Project management**
11. Open source, APIs, IPR
12. Embedded systems
13. Recap

Tampereen yliopisto
Tampere University

10. Project management

- overview of a project
- common problems (How to Fail)
- common success factors (Best Practices)
-
- tools
- scheduling / time estimations
- change management
- metrics / measurements (complexity, FPA, FiSMA, COCOMO,…)
- risk management
- indicators (e.g. ”traffic lights”)
-
- standards

Tampereen yliopisto
Tampere University

Current at course (w 46)

- **WE9 this week** (effort estimation)
- after those we think next week WE10 groups
- **continue updating your Trello (kanban) boards** = use at your process
- ***remember EXAM 2/3 (weeks 44-46, -17.11.)*; *diagrams (Dia), zip***
- **1st presentations group-to-group feedback is at PRP system**
- **EXAM 3/3 at weeks 47-49**

IT-Hekuma (companies recruiting) 13.11.2019 at Tietotalo and Sähkötalo.

Backlog items with deadline

• **09.09.2019**	at 23:59	Group forming (Moodle)
• **15.09.2019**	at 23:59	Trello creation (Trello)
• **13.10.2019**	at 23:59	Phase 1 documentation (Moodle)
• **13.10.2019**	at 23:59	Phase 1 presentation slides (PRP-tool)
• **Week 43**		Phase 1 presentations (Physical realm)
• **03.11.2019**	at 23:59	Phase 1 peer feedback (PRP-tool)
• **17.11.2019**	at 23:59	Phase 2 documentation (Moodle)
• **17.11.2019**	at 23:59	Phase 2 presentation slides (PRP-tool)
• **Week 47**		Phase 2 presentation (Physical realm)
• **01.12.2019**	at 23:59	Phase 2 peer feedback (PRP-tool)
• **08.12.2019**	at 23:59	Final delivery of project documentation (Moodle)
• **15.12.2019**	at 23:59	Final peer feedback and self assessments (PRP-tool).

Tampereen yliopisto
Tampere University

Weekly exercise attendees

	w36 WE1	w37 WE2	w38 WE3	w39 WE4	w40 WE5	w41 WE6	w44 WE7	w45 WE8	w46 WE9	w48 WE10
WED	0	14	9	5	8	9	9	7		
THU	21	13	14	17	16	13	6	17		

We will continue two Weekly Exercise groups, as long as the number of attendees are reasonable.

Tampereen yliopisto
Tampere University

Some famous PM (project management) books at top row.

I think at least 50 % of PM matters are valid for all application areas, so not depending which business or technical branch projects you have.

Tampereen yliopisto
Tampere University

Successful project = ?

How you define a "successful project" ? (FI: millainen on onnistunut projekti)

- in time
- within budget
- all required features done

The classic ones… not all happening often. **Why ?**

- customer is happy, well at least not angry, will do more business later
- **project workers are happy**
- project manager is happy
- real value to customer (at least some value, it is the **feeling**)
- project workers have learned… surely at least something
- project workers are proud of their work !
- or something else… a lot of more details may describe a successful project.

Think about that; successful project vs. successful product.

Successful project = ???

There is no one right answer for what is a ”successful” ICT project.

Every stakeholder on project has his/her own definition for ”success”.

Remember that when e.g. reading research statistics.

If there is not much what to measure with exact values, **you may consider the current project to previous ones**, or ”company average”.

Anyway, in every project you do LEARN something.

Successful project...

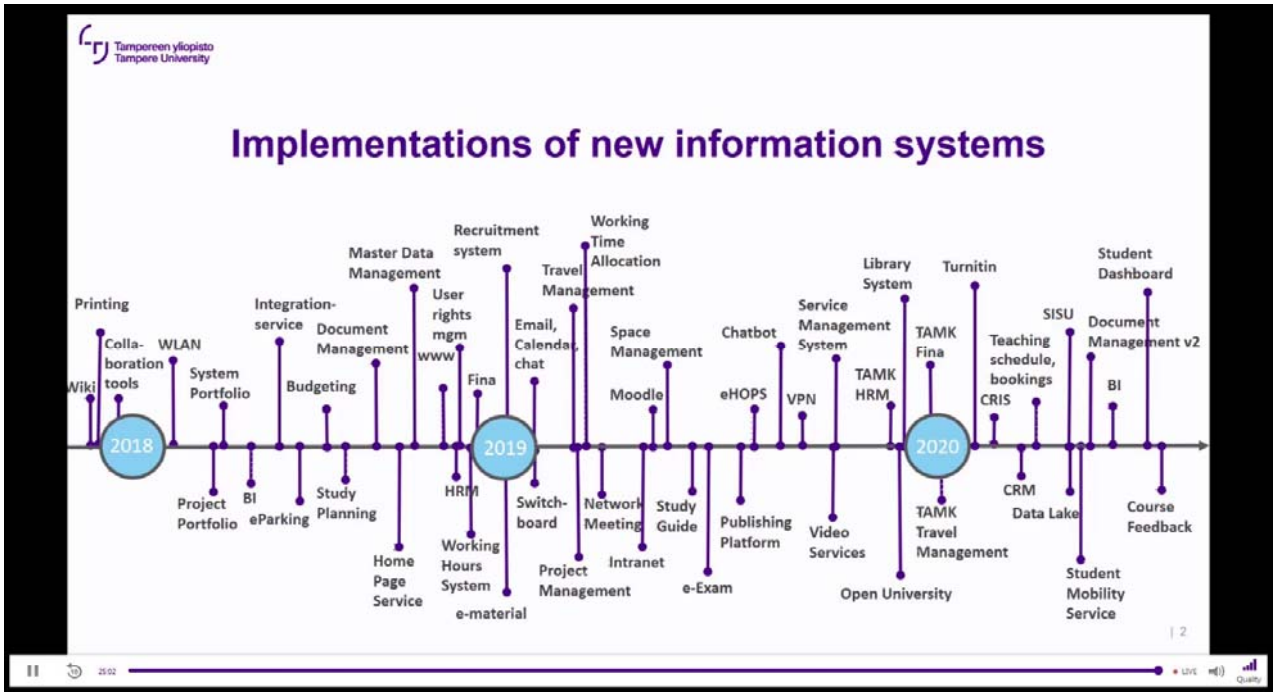
One important point is that from every project some kind of **Final Report** should be written, even a short (one page) one. It has ”**lessons learnt**”; **what went well** (and why), **what did not work** (and why), any **good tools, techniques or methods**. So new project managers can read at least what were the ”**best practices**” on previous projects. However, all projects are not similar, and all old rules may not apply to the next project. Successful ”best practices” may be formed to ”**patterns**” in company.

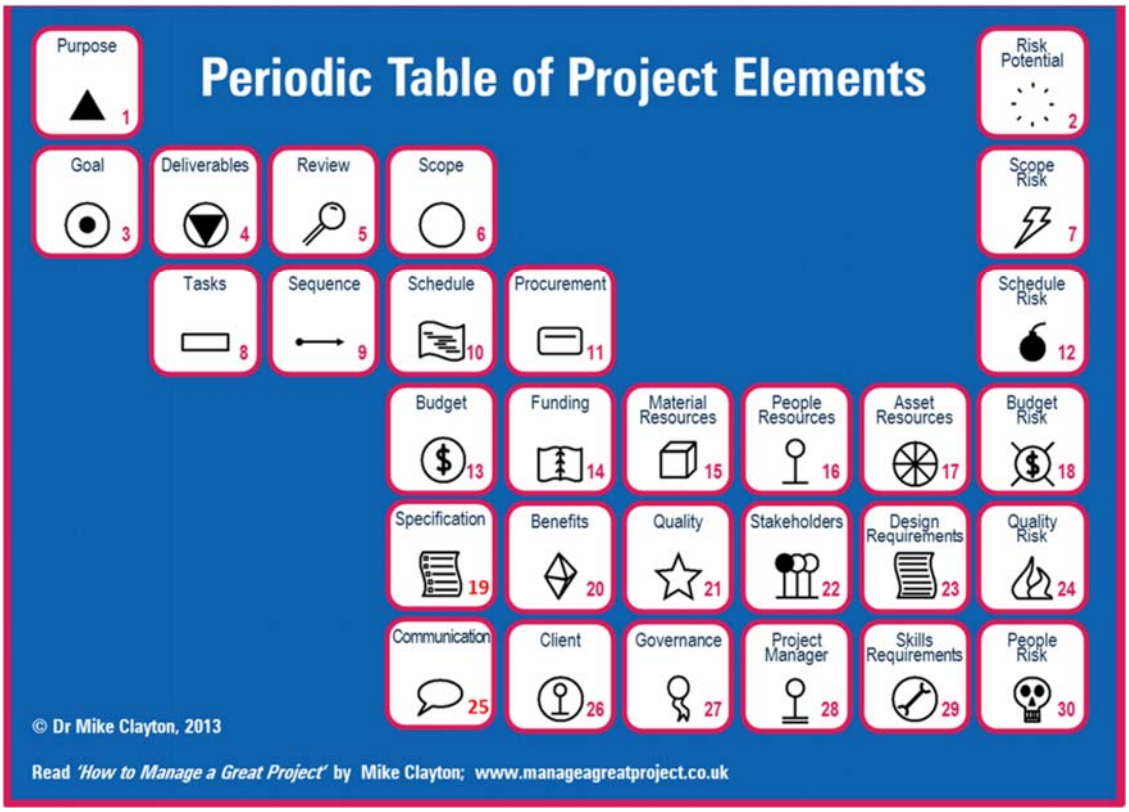
”Coffee table discussions” have tried to solve the old problem:

We know well what are the risks and failure reasons on software development projects. Why so many software projects still fail ? Are human beings stupid or ignorant ??

The only reason we have found, is: there are more and more new software projects established with new project managers, than current project managers get experience.

Big systems should be taken into use in small steps





Tampereen yliopisto
Tampere University

Management vs. leadership

Table 3-1. Team Management and Team Leadership Compared

Management	Leadership
Direct using positional power	Guide, influence, and collaborate using relational power
Maintain	Develop
Administrate	Innovate
Focus on systems and structure	Focus on relationships with people
Rely on control	Inspire trust
Focus on near-term goals	Focus on long-range vision
Ask how and when	Ask what and why
Focus on bottom line	Focus on the horizon
Accept status quo	Challenge status quo
Do things right	Do the right things
Focus on operational issues and problem solving	Focus on vision, alignment, motivation, and inspiration

[Guide to PMBOK, 2017]

Tampereen yliopisto
Tampere University

project management

Factors influencing project management

- Company size
- Software customers
- Software size
- Software type
- Organizational culture
- Software development processes
- These factors mean that project managers in different organizations may work in quite different ways.

Universal management activities

- ***Project planning***
  - Project managers are responsible for planning. estimating and scheduling project development and assigning people to tasks.
  - Covered in Chapter 23.
- ***Risk management***
  - Project managers assess the risks that may affect a project, monitor these risks and take action when problems arise.
- ***People management***
  - Project managers have to choose people for their team and establish ways of working that leads to effective team performance.

Management activities

- ***Reporting***
  - Project managers are usually responsible for reporting on the progress of a project to customers and to the managers of the company developing the software.
- ***Proposal writing***
  - The first stage in a software project may involve writing a proposal to win a contract to carry out an item of work. The proposal describes the objectives of the project and how it will be carried out.

Tampereen yliopisto
Tampere University

common problems

6point6 commissioned a survey of 300 CIOs in the UK and the US to examine their experiences of agile and measure how successfully the principles of agile are being applied and executed.

Chris Porter, CTO and co-founder of 6point6 and one of the authors of the report said: "Agile IT in the UK is facing a hidden crisis – **12% of agile projects are failing completely**.

You may fail, if you use agile "wrong".

[https://www.information-age.com/uk-wasting-37-billion-year-failed-agile-it-projects-123466089/]

"The Standish Group report 83.9% of IT projects partially or completely fail" [www.opendoorerp.com, 2019]

All of the top factors found in <u>failed</u> projects include:

- Incomplete Requirements
- Lack of user involvement
- Lack of resources
- Unrealistic expectations
- Lack of executive support
- Changing Requirements & Specifications
- Lack of planning
- Didn't need it any longer
- Lack of IT management
- Technical illiteracy.

Remember:
how you define "failed"... ?

Big/long projects surely are difficult to manage.

Q: How you eat an elephant ?
A: In small pieces.

Collected sw project data from USA

In the United States, we spend more than $250 billion each year on IT application development of approximately 175,000 projects.

The average cost of a development project

for a large company is $2,322,000;

for a medium company, it is $1,331,000; and

for a small company, it is $434,000.

A great many of these projects will fail. Software development projects are in chaos, and we can no longer imitate the three monkeys -- hear no failures, see no failures, speak no failures.!

The Standish Group research shows a staggering **31.1% of projects will be cancelled** before they ever get completed. Further results indicate 52.7% of projects will cost 189% of their original estimates.

One just has to look to the City of Denver to realise the extent of this problem. The failure to produce reliable software to handle baggage at the new Denver airport is costing the city $1.1 million per day.!

[Standish Group 2014]

common success factors

Tampereen yliopisto
Tampere University

Good team =

Factors of Success/Value	Points	Investment
Small Agile Projects	25	25%
Executive Sponsorship	15	20%
Emotional Maturity	15	20%
Talented Staff	10	15%
User Involvement	9	4%
Optimization	8	4%
SAME (Standard Architectural Management Environment)	6	3%
Modest Execution	5	3%
PM/Process Expertise	4	3%
Clear Business Objectives	3	3%
Total Points & Yearly Investment	100	100%

[Standish Group, 2016]

Tampereen yliopisto
Tampere University

Project Success Factors	% of Responses
1. User Involvement	15.9%
2. Executive Management Support	13.9%
3. Clear Statement of Requirements	13.0%
4. Proper Planning	9.6%
5. Realistic Expectations	8.2%
6. Smaller Project Milestones	7.7%
7. Competent Staff	7.2%
8. Ownership	5.3%
9. Clear Vision & Objectives	2.9%
10. Hard-Working, Focused Staff	2.4%
Other	13.9%

[CHAOS report, 2015]

Tampereen yliopisto
Tampere University

The definitions for these factors are:

Executive Support: when an executive or group of executives agrees to provide both financial and emotional backing. The executive or executives will encourage and assist in the successful completion of the project.

Emotional maturity is the collection of basic behaviors of how people work together. In any group, organization, or company it is both the sum of their skills and the weakest link that determine the level of emotional maturity.

User Involvement: takes place when users are involved in the project decision-making and information-gathering process. This also includes user feedback, requirements review, basic research, prototyping, and other consensus-building tools.

Optimization is a structured means of improving business effectiveness and optimizing a collection of many small projects or major requirements. Optimization starts with managing scope based on relative business value.

Skilled staff are people who understand both the business and the technology. A skilled staff is highly proficient in the execution of the project's requirements and deliver of the project or product.

CHAOS FACTORS OF SUCCESS

FACTORS OF SUCCESS	POINTS	INVESTMENT
Executive Sponsorship	15	15%
Emotional Maturity	15	15%
User Involvement	15	15%
Optimization	15	15%
Skilled Resources	10	10%
Standard Architecture	8	8%
Agile Process	7	7%
Modest Execution	6	6%
Project Management Expertise	5	5%
Clear Business Objectives	4	4%

[Standish Group, 2015]

Tampereen yliopisto
Tampere University

Standish group [via www.opendoorerp.com]

The top five factors found in successful projects are:

User involvement

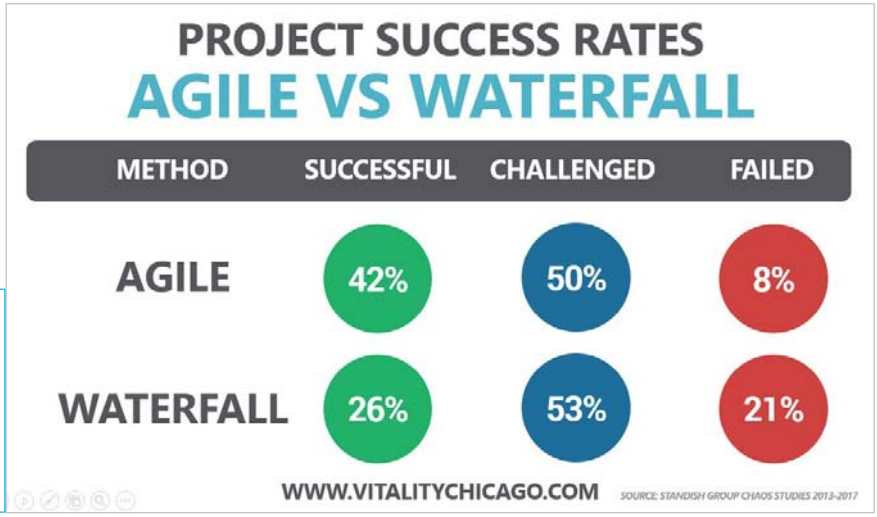
Executive management support

Clear Statement of Requirements

Proper planning

Realistic expectations.

There is still much room for improvements; less than half of agile projects were "successful".



Tampereen yliopisto
Tampere University

Success factors were:

1. Good User Participation/ Involvement.
2. Good Requirements and Specifications.
3. Good of Skills.
4. Complete Requirements.
5. Good Management and Performance from Vendor/ Contractor
6. Good of Project Planning.
7. Realistic Expectations.
8. Good of Resources.
9. Good of Executive Support (Top Management).

Table 3. Frequency Of Failure Factors Selected By Respondents

Failure factors	Frequency	Percentage	Ranking
Lack of User Involvement	38	79.17	1
Changing Requirements and Specifications	36	75.00	2
Lack of Skills	35	72.92	3
Incomplete Requirements	34	70.32	4
Vendor/ Contractor Performance and Management	32	66.67	5
Lack of Planning	30	62.50	6
Unrealistic Expectations	29	60.42	7
Lack of Resources	27	56.25	8
Lack of Executive Support (Top Management	24	50.00	9
Project Team Turn Over	18	37.50	10
Lack of Communication among project team	18	37.50	11
Project Complexity	18	37.50	12
Lack of User Knowledge	17	35.42	13
Lack of IT Management	16	33.33	14
Technology Illiteracy	15	31.25	15
Inter Department Communication	15	31.25	16

[Shamsudin Md Sarif et. Al.: Investigation of Success and Failure Factors in IT Project Management, 2018]

Tampereen yliopisto
Tampere University

project working skills

By the way, are project workers human beings, or just ”resources” ?

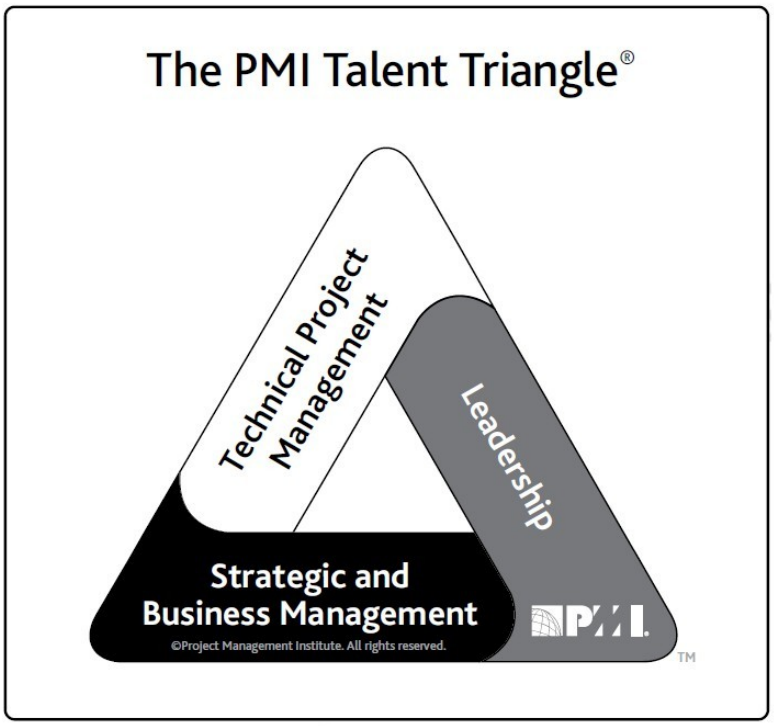


Figure 3-2. The PMI Talent Triangle®

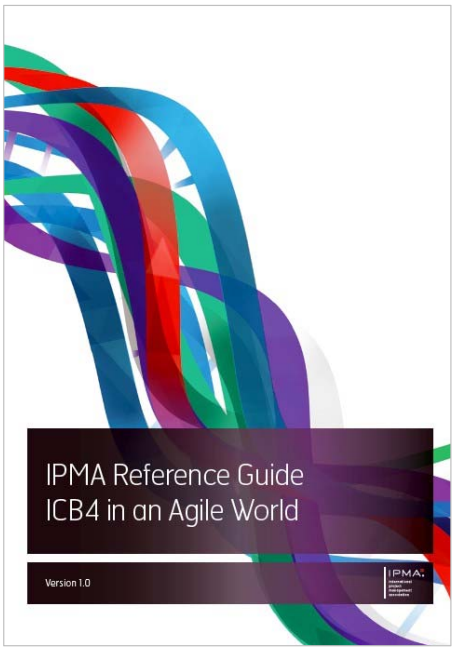
PMI = Project Management Institute

”USA”

IPMA ICB4,
individual competence baseline

IPMA = International Project Management Association

”Europe”



		Knowledge	Comprehension	Application	Analysis	Synthesis	Evaluation
Perspective	Strategy						
	Governance, Structures and Processes						
	Compliance, Standards and Regulations						
	Power and Interest						
	Culture and Values						
People	Self-reflection and Self-management						
	Personal Integrity and Reliability						
	Personal Communication						
	Relations and Engagement						
	Leadership						
	Teamwork						
	Conflict and Crisis						
	Resourcefulness						
	Negotiation						
	Results orientation						
Practice	Design						
	Goals and requirements						
	Scope						
	Time						
	Organisation and Information						
	Quality						
	Finance						
	Resources						
	Procurement						
	Plan and Control						
	Risk and Opportunity						
	Stakeholders						
	Change and Transformation						

IPMA Reference guide ICB4 in an Agile World | 41

Tampereen yliopisto
Tampere University

tools

Tampereen yliopisto
Tampere University

tools

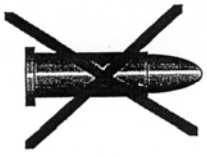
There are a lot of different project management tools available.

Mostly those are **visualising tools** (Gantt, BurnUp, BurnDown,…), where you should input dates and amounts of effort. So the **tools do not plan the project for you**.

According to experienced Project Managers, the best tools are

- **coffee room table**
- **pen and paper**
- **spreadsheet.**

There is no silver bullet!



No tool, no method will save you from having to think!

Tampereen yliopisto
Tampere University

scheduling, time estimations

Tampereen yliopisto
Tampere University

scheduling, time estimations

Estimating project variables is more or less **guessing** (”questimation”).

If you have **statistics** from similar previous projects, you get some idea of the numbers (”about that much”).

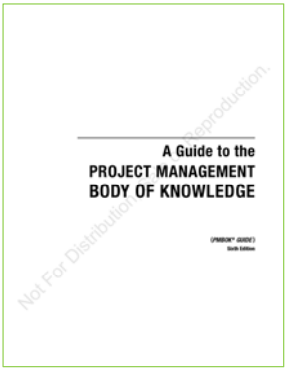
You may ask experienced **colleague** for his/her estimation (”academic guess”).

You may gather together with colleagues and discuss about each estimated matter. You may use Planning Poker here, too.

New project managers typically do longer and detailed estimations, while experienced project managers do shorter and more general estimations (as by their working experience).

scheduling, time estimations

Figure 6-1. Project Schedule Management Overview



[Guide to PMBOK, 2017]

scheduling

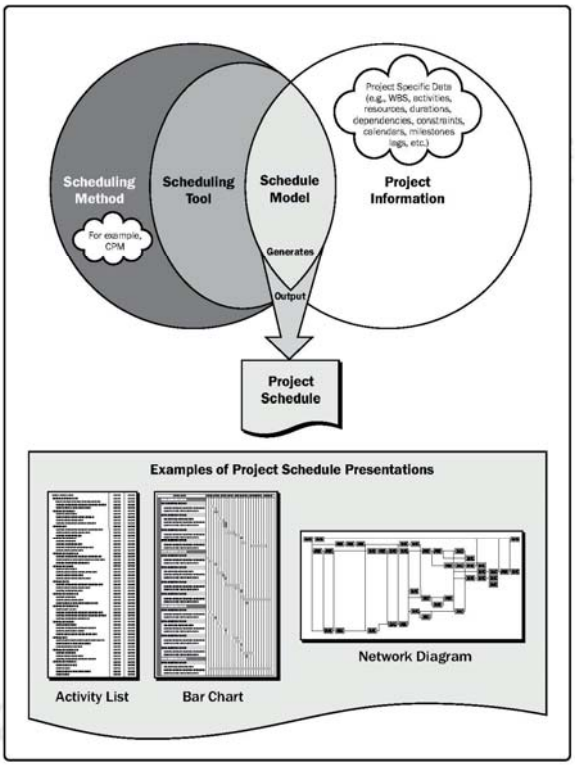


Figure 6-2. Scheduling Overview

[Guide to PMBOK, 2017]

Project scheduling

- Project scheduling is the process of **deciding how the work in a project will be organized** as separate tasks, and **when** and **how** these tasks will be executed.
- You **estimate** the calendar time needed to complete each task, the effort required and who will work on the tasks that have been identified.
- You also have to estimate the resources needed to complete each task, such as the disk space required on a server, the time required on specialized hardware, such as a simulator, and what the travel budget will be.

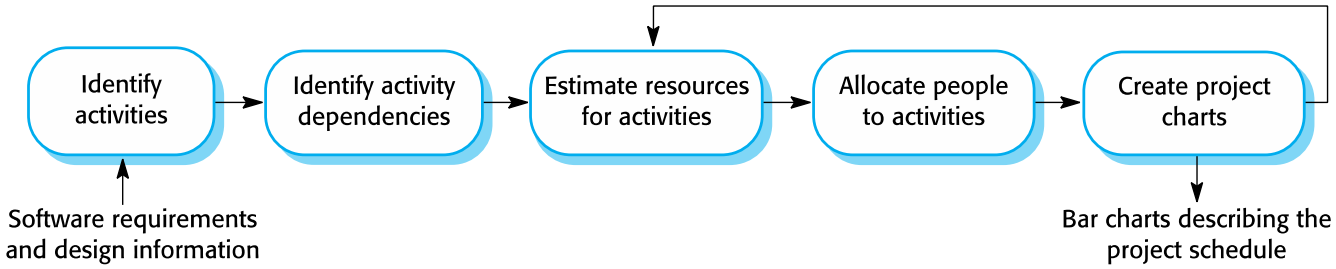
Whatever you do for the first time, is difficult.
At second time it goes better and much easier.

Project scheduling activities

- **Split** project into tasks and estimate time and resources required to complete each task.
- **Organize** tasks concurrently to make optimal use of workforce.
- **Minimize** task dependencies to avoid delays caused by one task waiting for another to complete.
- Dependent on project managers intuition and experience.

You may use top-dow or bottom-up estimation for tasks.

The project scheduling process



Scheduling problems

- Estimating the **difficulty** of problems and hence the **cost** of developing a solution is hard.
- Productivity is not proportional to the number of people working on a task.
- **Adding people to a late project makes it later because of communication overheads**.
- **The unexpected always happens**. Always allow contingency in planning.

Experienced Project Managers spare one weekday at their calendar for unplanned tasks, as a ”buffer” (20 % of weekly working time).

Schedule presentation

- Graphical notations are normally used to illustrate the project schedule.
- These show the project breakdown into tasks. Tasks should not be too small. They should take about a week or two.
- Calendar-based
  - Bar charts are the most commonly used representation for project schedules. They show the schedule as activities or resources against time.
- Activity networks
  - Show task dependencies

WBS = Work Breakdown Structure, split work to manageable small parts.

Tampereen yliopisto
Tampere University

change management

There will be smaller or bigger **changes and surprises** in every project, so why plan anything at all ?

Surely there will be changes in every project, but the overall goals and techniques are known (planned). Experienced project managers can handle changes.

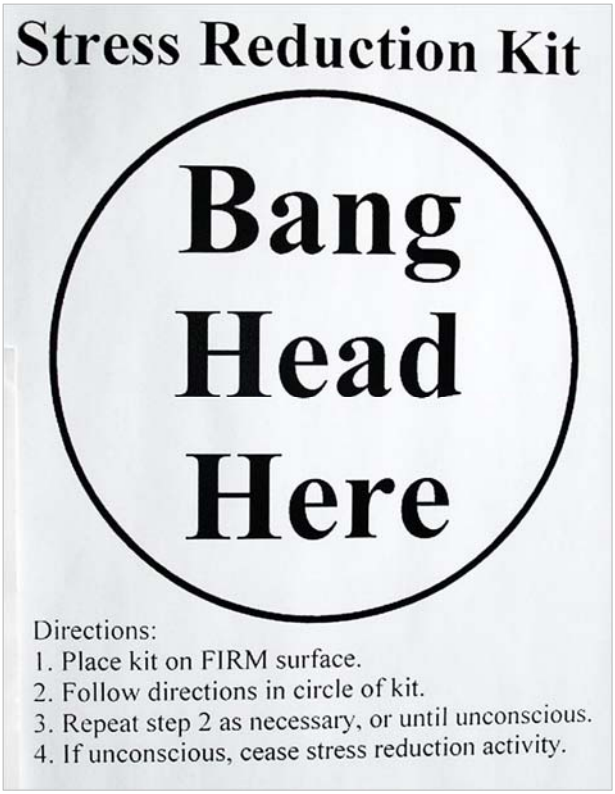
Well, if you change (stretch) the ”iron triangle” (fast – cheap - good), remember to **update** Project Plan also.

Changes should be **documented**, and should become **known** for those who it may concern.

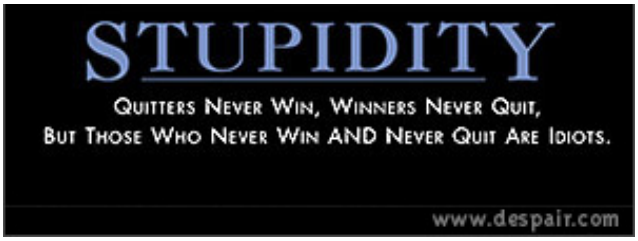
”How do you want your software ?
FAST - CHEAP – GOOD, pick two.”

[hackernoon.com]

This SRK was made mainly for project managers, but it works also for other project personnel and stakeholders.



Sometimes project termination or cancellation would be a good option.



Big Room (FI: allianssimalli)

- An effective **Big Room** supports **cross-functional team collaboration** by advancing work and bringing the larger team up to speed on the activities of other groups or individuals. It allows **teams to understand their impact across clusters or work groups**. The Big Room also provides teams with the **time to discuss project-wide concerns** like budgets, hot topics, or global changes. The term Big Room refers more to the **behaviors and actions** of the team than the physical space. The Big Room is more than co-location of people; it is about **collaborative behavior** and the work they are producing.

[leanconstruction.org]

project documentation checklist

Don’t be scared, this is for heavy/big projects.

But it is a good **checklist** that you forget nothing.

Table 4-1. Project Management Plan and Project Documents

Project Management Plan	Project Documents	
1. Scope management plan	1. Activity attributes	19. Quality control measurements
2. Requirements management plan	2. Activity list	20. Quality metrics
3. Schedule management plan	3. Assumption log	21. Quality report
4. Cost management plan	4. Basis of estimates	22. Requirements documentation
5. Quality management plan	5. Change log	23. Requirements traceability matrix
6. Resource management plan	6. Cost estimates	24. Resource breakdown structure
7. Communications management plan	7. Cost forecasts	25. Resource calendars
8. Risk management plan	8. Duration estimates	26. Resource requirements
9. Procurement management plan	9. Issue log	27. Risk register
10. Stakeholder engagement plan	10. Lessons learned register	28. Risk report
11. Change management plan	11. Milestone list	29. Schedule data
12. Configuration management plan	12. Physical resource assignments	30. Schedule forecasts
13. Scope baseline	13. Project calendars	31. Stakeholder register
14. Schedule baseline	14. Project communications	32. Team charter
15. Cost baseline	15. Project schedule	33. Test and evaluation documents
16. Performance measurement baseline	16. Project schedule network diagram	
17. Project life cycle description	17. Project scope statement	
18. Development approach	18. Project team assignments	

[Guide to PMBOK, 2017]

metrics

Tampereen yliopisto
Tampere University

metrics

Traditionally metrics in software projects have been based on **estimating required working hours**; from that it was possible to estimate cost and schedule.

Some famous metrics are

- COCOMO, COCOMO II = COnstructive COst MOdel
- FPA = Function Point Analysis
- FiSMA FPA = Function Point Analysis by Finnish Software Metrics Association
- and there are much more…

Nowadays the ”best” metrics may well be Project Manager’s and Developer Team’s experience.

Tampereen yliopisto
Tampere University

metrics

COCOMO, COCOMO II (2000)

- includes formulas for calculating required effort and thus schedule
- based on estimating system’s LOC (lines of code), huge amount of project statistics
- many variables according organisation; e.g. complexity and skills
- naturally depends also from programming language.

FPA = Function Point Analysis

- system’s functions are counted
- function points can be converted to LOC.

FiSMA FPA = Function Point Analysis by Finnish Software Metrics Association

Tampereen yliopisto
Tampere University

Some metrics www calculators

UofSC www calculator for COCOMO II

https://csse.usc.edu/tools/COCOMOII.php

NASA STRS www page for COCOMO calculation

https://strs.grc.nasa.gov/repository/forms/cocomo-calculation/

UofM Basic COCOMO calculators

http://groups.umd.umich.edu/cis/tinytools/cocomo.html

http://www.edtechnology.in/cocomo/

UofM Function Point calculator

http://groups.umd.umich.edu/cis/course.des/cis375/projects/fp99/main.html

Tampereen yliopisto
Tampere University

Some COCOMO www calculators

UofSC www calculator for COCOMO II

https://csse.usc.edu/tools/COCOMOII.php

NASA STRS www page for COCOMO calculation

https://strs.grc.nasa.gov/repository/forms/cocomo-calculation/

UofM Basic COCOMO calculators

http://groups.umd.umich.edu/cis/tinytools/cocomo.html

http://www.edtechnology.in/cocomo/

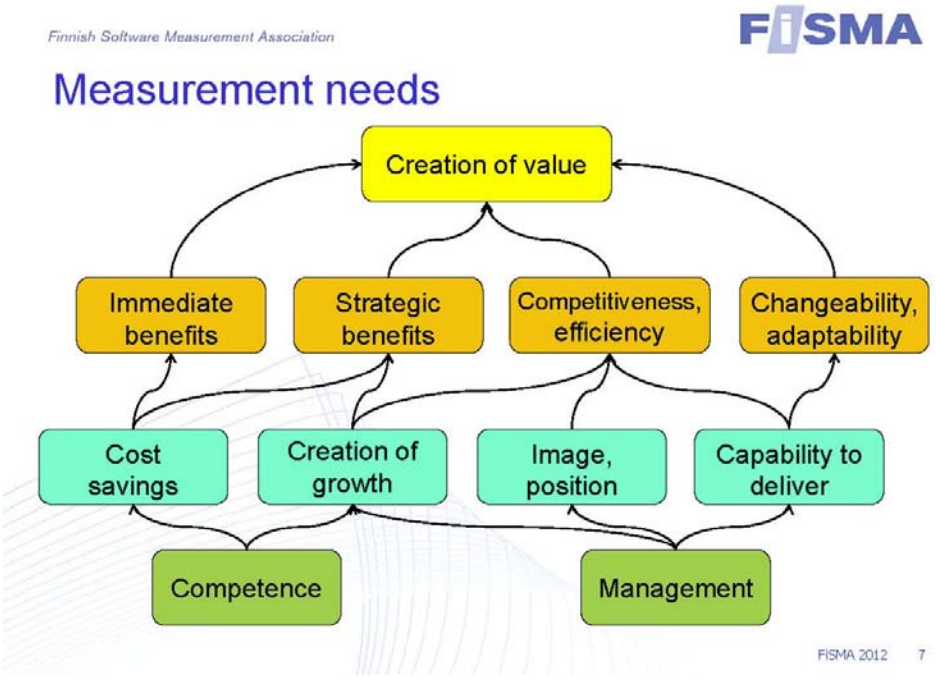
Some Function Point www calculators

UofM Function Point calculator
http://groups.umd.umich.edu/cis/course.des/cis375/projects/fp99/main.html

UofM FP calculator
http://groups.umd.umich.edu/cis/course.des/cis525/js/f00/harvey/FP_Calc.html

JMU Function Point calculator
https://w3.cs.jmu.edu/bernstdh/web/common/webapps/oop/fpcalculator/FunctionPointCalculator.html

FiSMA = Finnish Software Measurement Association



FiSMA

ICS > 35 > 35.080

ISO/IEC 29881:2010

Information technology — Systems and software engineering — FiSMA 1.1 functional size measurement method

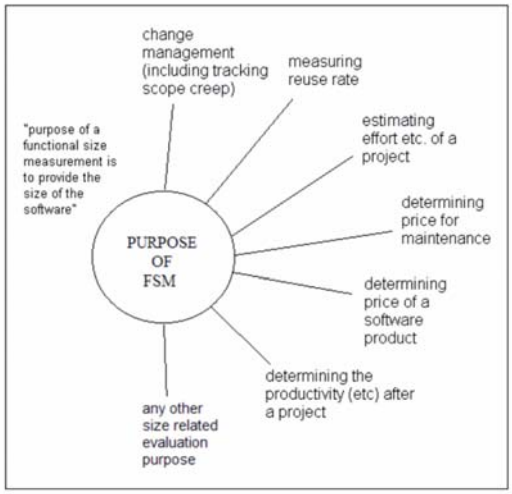


Figure 2 — Common purposes of Functional Size Measurement

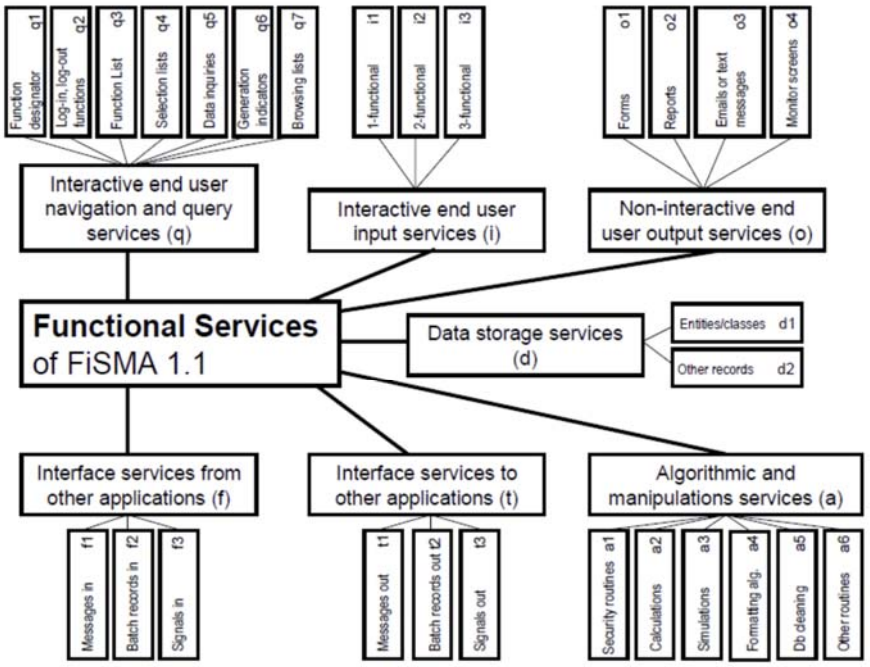
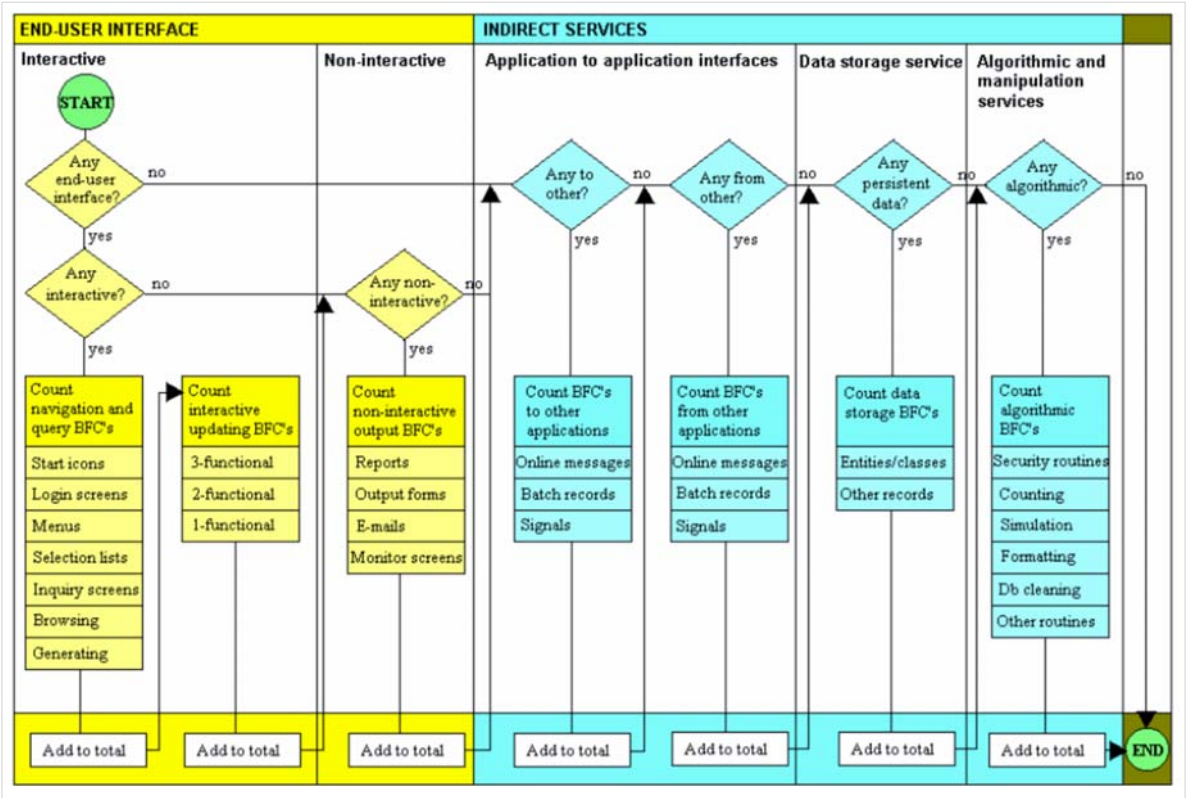


Figure 3 — FiSMA 1.1 BFC classes and BFC types

BFC = base functional component

FiSMA process



Finnish Software Measurement Association

FiSMA

Condition based classification

- Typical areas of software metrics are:
  - Software product (itself)
  - The process used to produce software
  - The management of developing software
  - Leading and managing software business
- Other classifiers:
  - The software product standard: internal, external, in use
  - The software process standard: capability, maturity
  - Software supplier / customer organisation / end-user
  - Project model: cost, time, quality, resources, workload, benefit/profit
  - Critical systems: stability, integrity, method conformance
  - Life cycle model: specification, technical planning, development, verification, validation, production (in all e.g. V&V findings, coverage, traceability)
  - Maturity wise: initial, managed, performed sets

FiSMA 2012 5

COSMIC

COSMIC Definition

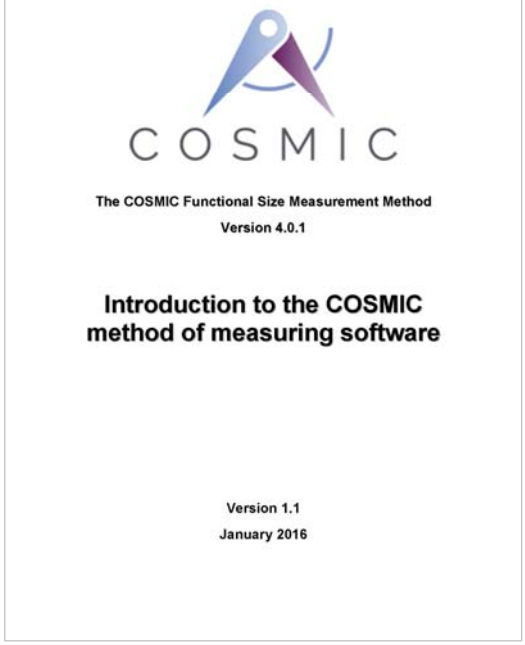
3

COSMIC stands for the Common Software Measurement International Consortium. COSMIC is a voluntary organization that has defined an open, ISO standard functional size measurement (FSM[1]) method, based on fundamental software engineering principles.

The International Standards Organization (ISO) FSM standards are:

COSMIC (ISO, ISO/IEC 19761)
IFPUG (ISO, ISO/IEC 20926)
Mark-II (ISO, ISO/IEC 20968)
NESMA (ISO, ISO/IEC 24570)
FiSMA (ISO, ISO/IEC 29881)

ePlayer 3 / 39



ICS › 35 › 35.080

ISO/IEC 14143-6:2012

Information technology — Software measurement — Functional size measurement — Part 6: Guide for use of ISO/IEC 14143 series and related International Standards

THIS STANDARD WAS LAST REVIEWED AND CONFIRMED IN 2019. THEREFORE THIS VERSION REMAINS CURRENT.

ICS › 35 › 35.080

ISO/IEC 20968:2002

Software engineering — Mk II Function Point Analysis — Counting Practices Manual

THIS STANDARD WAS LAST REVIEWED AND CONFIRMED IN 2008. THEREFORE THIS VERSION REMAINS CURRENT.

ICS › 35 › 35.080

ISO/IEC 19761:2011

Software engineering — COSMIC: a functional size measurement method

THIS STANDARD WAS LAST REVIEWED AND CONFIRMED IN 2019. THEREFORE THIS VERSION REMAINS CURRENT.

ICS › 35 › 35.080

ISO/IEC 24570:2018

Software engineering — NESMA functional size measurement method — Definitions and counting guidelines for the application of function point analysis

ICS › 35 › 35.080

ISO/IEC 20926:2009

Software and systems engineering — Software measurement — IFPUG functional size measurement method 2009

THIS STANDARD WAS LAST REVIEWED AND CONFIRMED IN 2019. THEREFORE THIS VERSION REMAINS CURRENT.

ICS › 35 › 35.080

ISO/IEC 29881:2010

Information technology — Systems and software engineering — FiSMA 1.1 functional size measurement method

THIS STANDARD WAS LAST REVIEWED AND CONFIRMED IN 2019. THEREFORE THIS VERSION REMAINS CURRENT.

NIST IR 8101

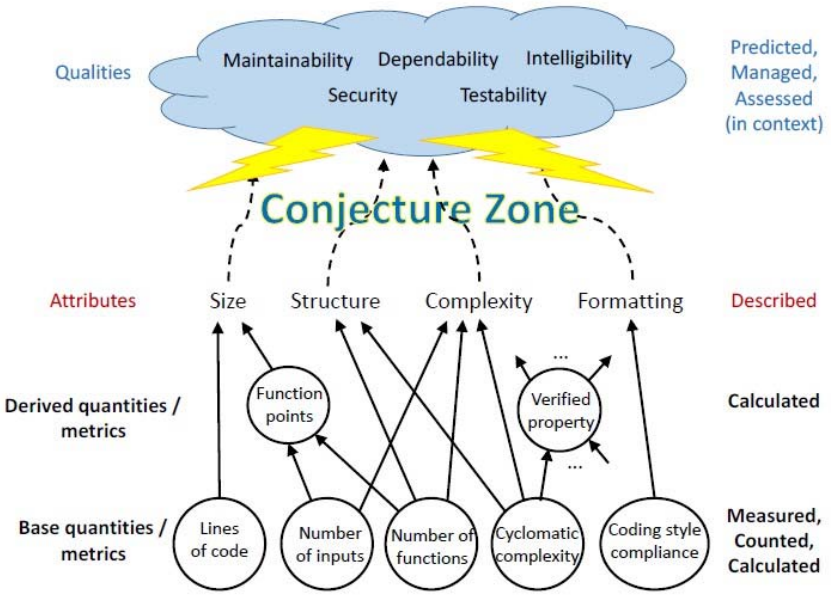


Figure 4: Relating qualities to quantities.

software metrics [Tutorialspoint]

Software metrics can be classified into three categories –

Product metrics – Describes the characteristics of the product such as size, complexity, design features, performance, and quality level.

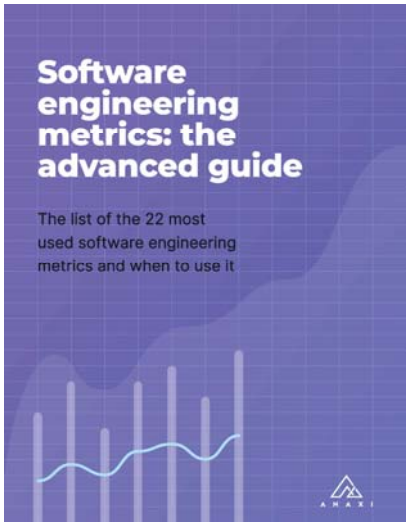
Process metrics – These characteristics can be used to improve the development and maintenance activities of the software.

Project metrics – This metrics describe the project characteristics and execution. Examples include the number of software developers, the staffing pattern over the life cycle of the software, cost, schedule, and productivity.

If you want to know how a project is going, you have to measure something.
Select the metrics wisely (measurable exact values) and reasonably (which matters).

Perhaps it is not reasonable to measure computer processor load, nor worker's blood pressure.

JAC (just another classification) of metrics



What are software engineering metrics?

In software, there are 2 categories of metrics and we use different names for those:

1. Software engineering metrics, also known as software development metrics, or software delivery performance, every team has a different name for them, it seems. What is important here is that those indicators measure how software is being built and the engineering team productivity.

2. Software or application performance metrics are the metrics of the software delivered, response time of the application, etc. NewRelic is typically one of the main providers of such metrics. You could also think of this in terms of customer satisfaction metrics - Net Promoter Score typically).

[https://anaxi.com/software-engineering-metrics-an-advanced-guide/]

Top 5 Software Metrics to Manage Development Projects

Types of Software Metrics

- **Formal code metrics**—Such as Lines of Code (LOC), code complexity, Instruction Path Length, etc. In modern development environments, these are considered less useful.
- **Developer productivity metrics**—Such as active days, assignment scope, efficiency and code churn. These metrics can help you understand how much time and work developers are investing in a software project.
- **Agile process metrics**—Such as lead time, cycle time and velocity. They measure the progress of a dev team in producing working, shipping-quality software features.
- **Operational metrics**—Such as Mean Time Between Failures (MTBF) and Mean Time to Recover (MTTR). This checks how software is running in production and how effective operations staff are at maintaining it.
- **Test metrics**—Such as code coverage, percent of automated tests, and defects in production. This measures how comprehensively a system is tested, which should be correlated with software quality.
- **Customer satisfaction**—Such as Net Promoter Score (NPS), Customer Effort Score (CES) and Customer Satisfaction Score (CSAT). The ultimate measurement of how customers experience the software and their interaction with the software vendor.

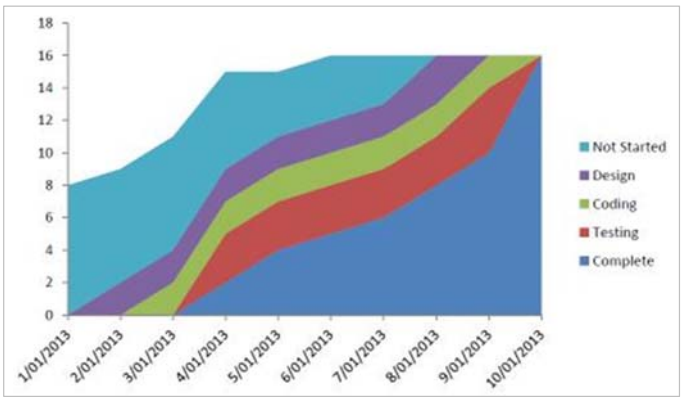
[https://www.sealights.io/software-development-metrics/top-5-software-metrics-to-manage-development-projects-effectively/]

Tampereen yliopisto
Tampere University

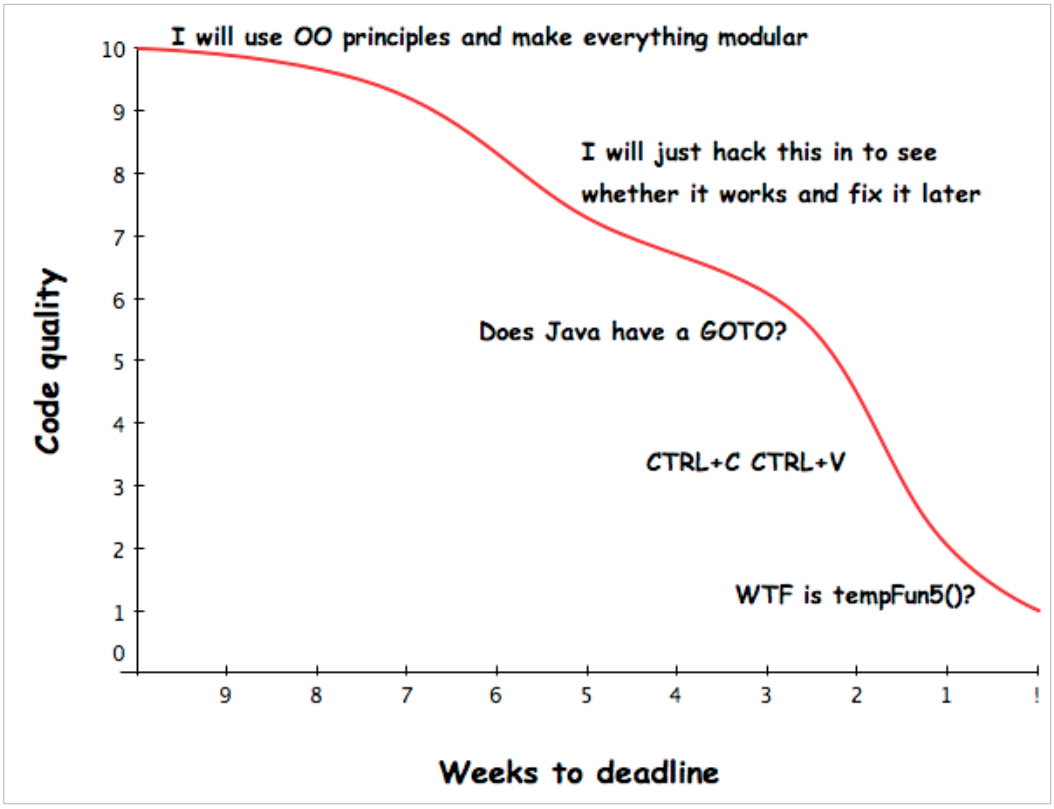
Actually in Finland many many years ago one company had this kind of ”bug bonus”, but just for a few months.

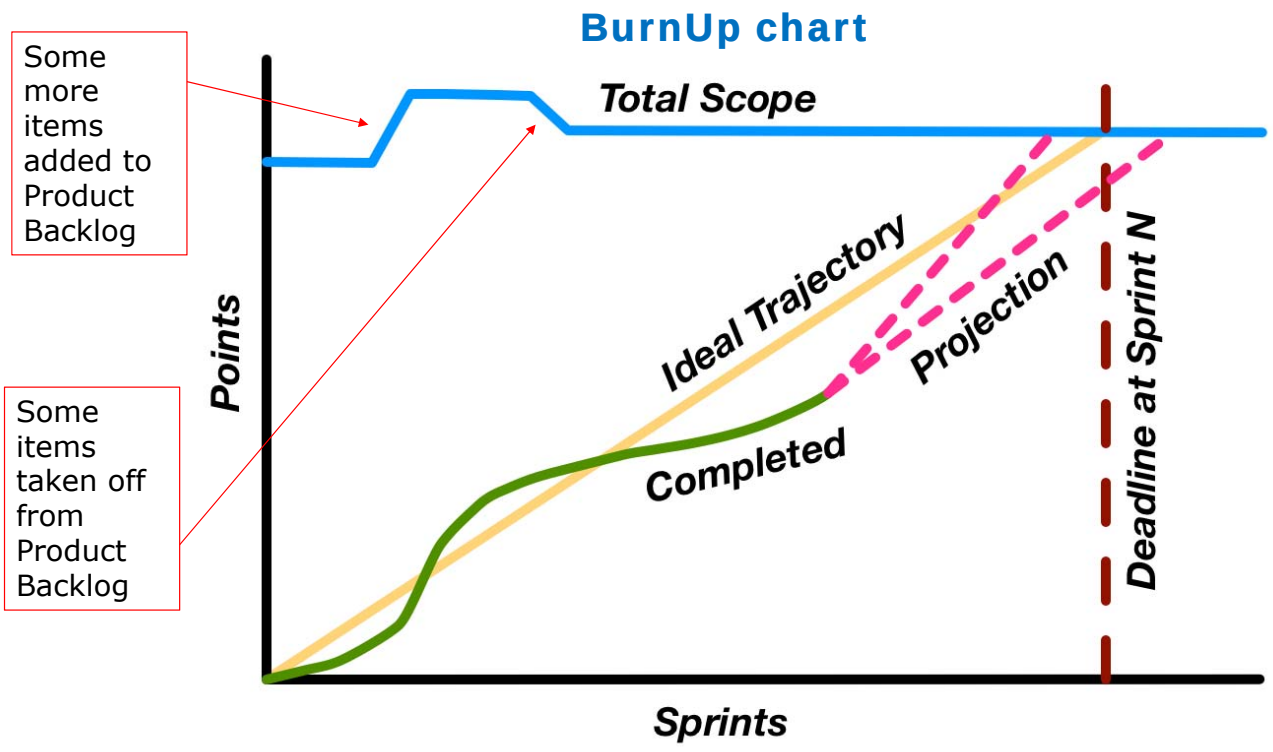
Tampereen yliopisto
Tampere University

Some sw project metrics graphs

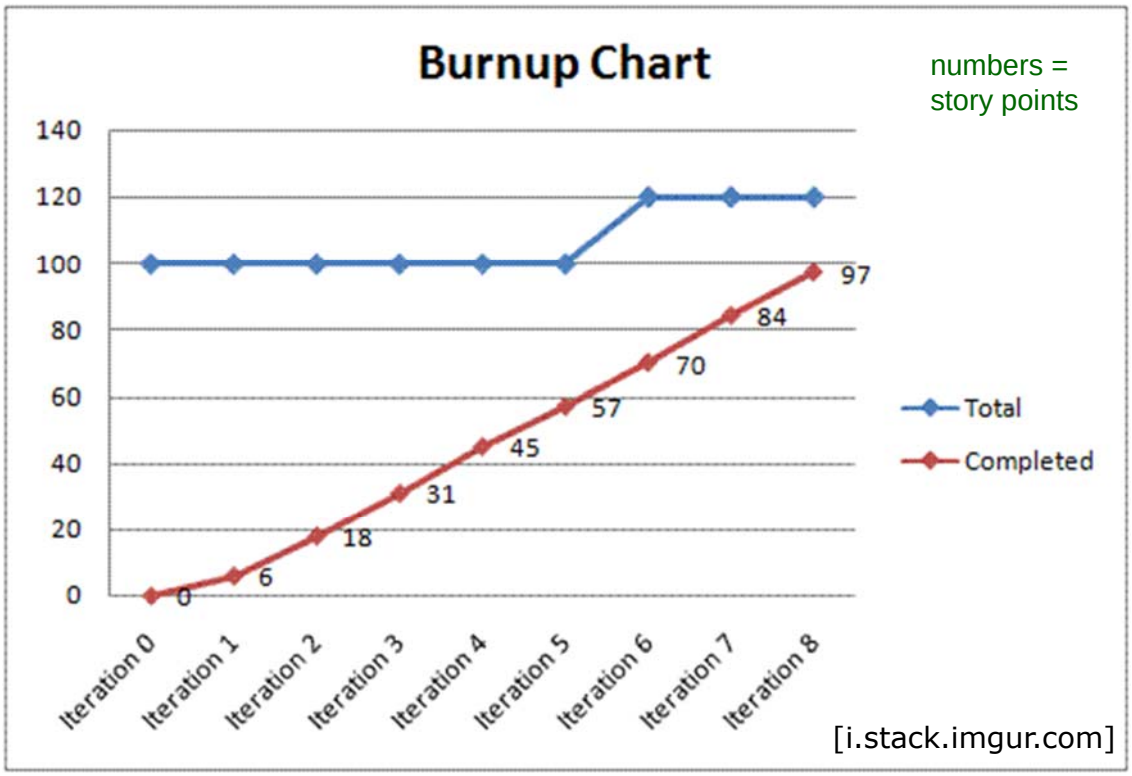


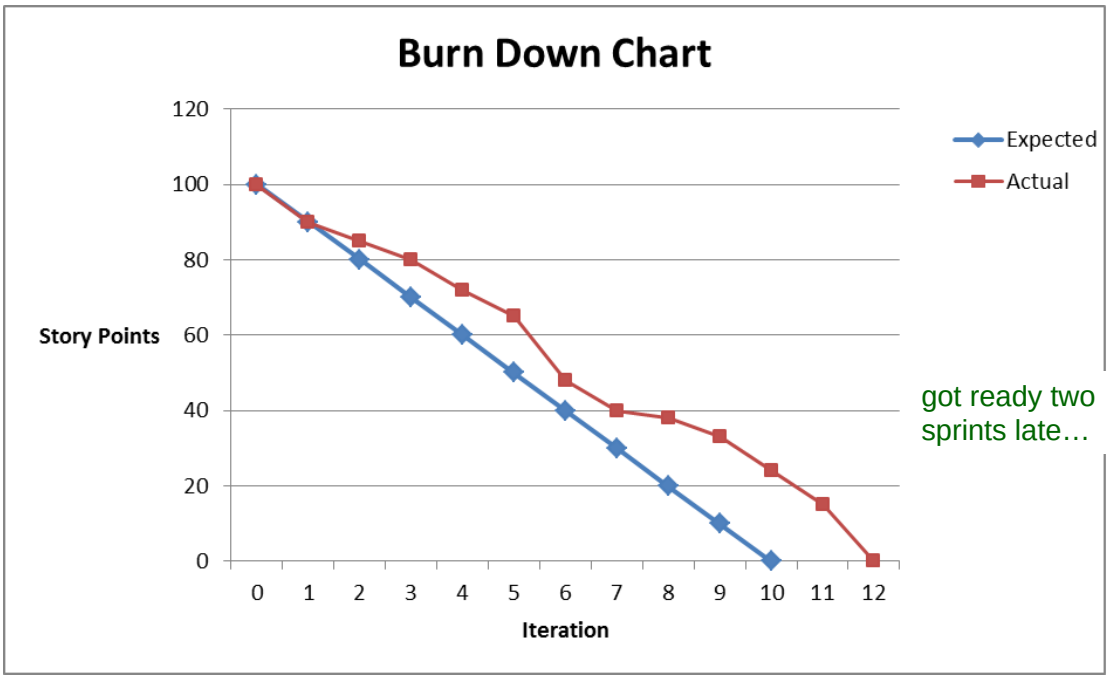
[https://www.daxx.com/blog/development-trends/top-software-development-metrics]



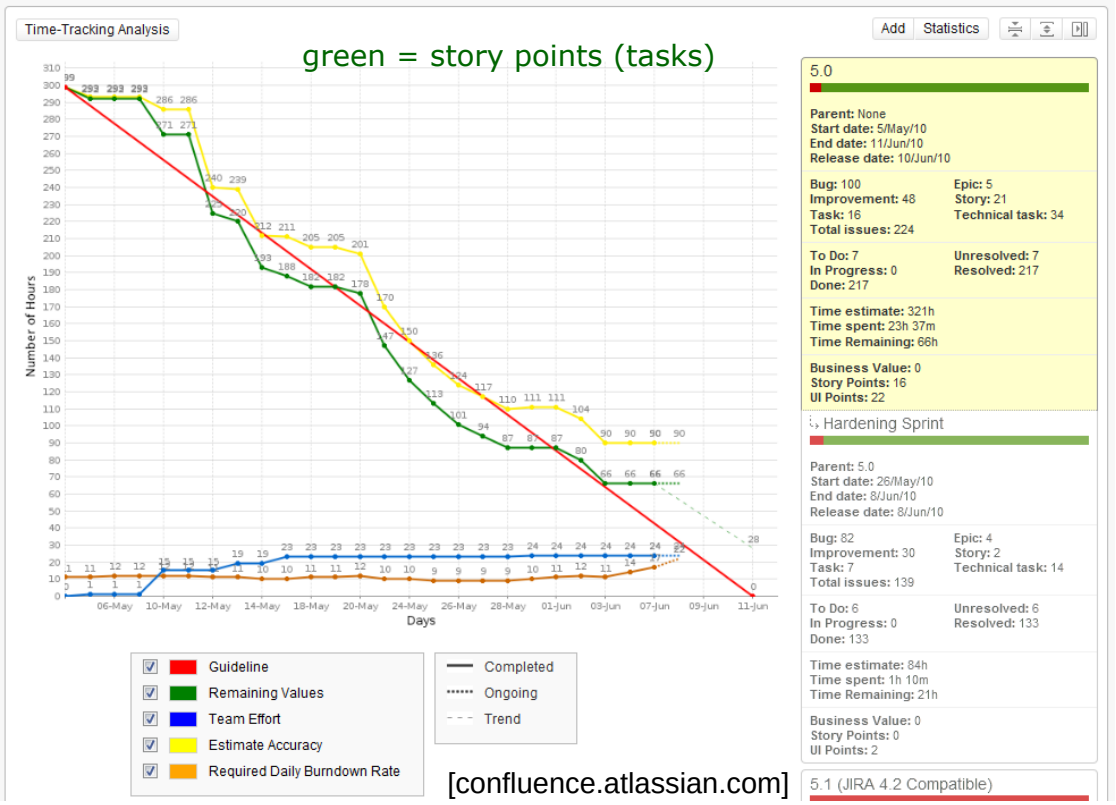


[spin.atomicobject.com]

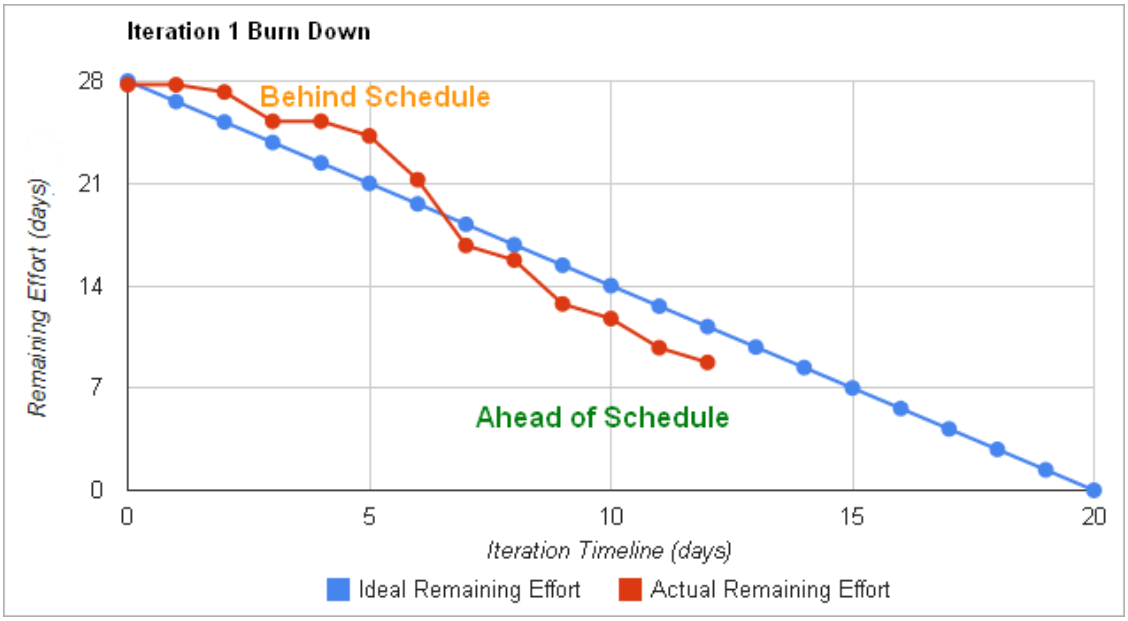




[blog.caplin.com]



[confluence.atlassian.com]



[i.stack.imgur.com]

Tampereen yliopisto
Tampere University

Some effort estimation techniques

Table 7. SDEE techniques potentially suitable for small software organizations.

Techniques	Classificassion Dimensions	Licence			Data				Parametric				Expert			Model			Analogy			Regression			Composite			Artificial Intelligence		
	Techniques / Values	Proprietary	Non-Proprietary	Unknown	Many data	Sparse Data	No Data	Unknown	Parametric	Non-Parametric	Semi-Parametric	Unknown	Need Expert	No Need Expert	Unknown	Model-Based	No Model-Based	Unknown	Analogy-Based	No Analogy-Based	Unknown	Regression-Based	No Regression-Based	Unknown	Composite Technique	Single Technique	Unknown	AI-Based	No AI-Based	Unknown
	Delphi		x				x			x			x				x			x			x			x			x	
	Work Breakdown Structure (WBS)		x				x			x			x				x			x			x			x			x	
	Rule Based			x		x						x	x					x			x			x			x				x
	Guesstimation		x				x			x			x				x			x			x			x			x	
	Estimeeting		x			x				x			x				x		x				x			x			x	
	Planning Game		x				x			x			x				x			x			x			x			x	
	Expert Judgment		x				x			x			x				x			x			x			x			x	
	Analytic Hierarchy Process (AHP)		x			x				x			x				x			x			x			x			x	
	Planning Poker		x				x			x			x				x			x			x			x			x	
	Constructive Agile Estimation Algorithm		x			x			x				x			x				x		x			x				x	
	COBRA – Cost Estimation Benchmarking and Risk Analysis	x				x			x				x			x				x					x				x	
	Colaborative Filtering		x			x					x		x				x		x					x			x				x
	Use Case Points (UCP)		x			x					x		x			x			x				x			x				x	
	UCP Method Modification		x			x					x		x			x			x				x			x				x	
	Bagging + MSP/Model trees (MT)		x			x						x		x		x				x			x			x		x			
	Random + MLP		x			x						x		x		x				x			x			x		x			
	Bagging + Adaptive neuro fuzzy inference system (ANFIS)		x			x						x		x		x				x		x			x				x		
	SUMMARY	17	101	1	102	11	6	0	35	36	19	29	20	98	1	97	20	2	16	91	12	36	59	24	70	32	17	37	63	19
		119			119				119				119			119			119			119			119			119		

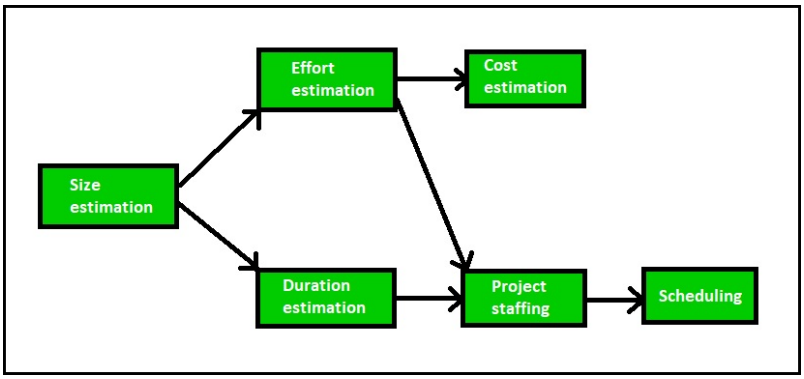
[Vera et.al.: Survey of Software Development Effort Estimation Taxonomies]

Tampereen yliopisto
Tampere University

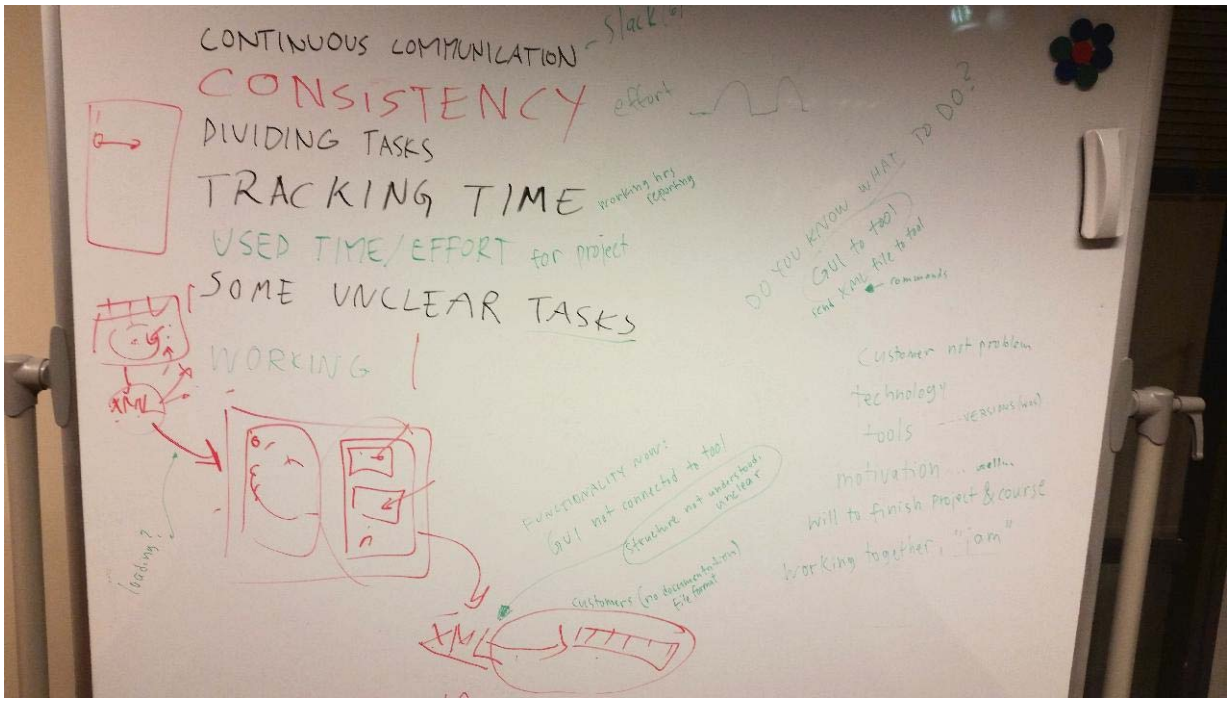
project monitoring and control

Project Estimation: Project Size Estimation is the most important parameter based on which all other estimations like cost, duration and effort are made.

[https://www.geeksforgeeks.org/software-engineering-role-and-responsibilities-of-a-software-project-manager/]



- **Scheduling:** After completion of estimation of all the project parameters, scheduling for manpower and other resources are done.
- **Staffing:** Team structure and staffing plans are made.
- **Risk Management:** The project manager should identify the unanticipated risks that may occur during project development risk, analysis the damage might cause these risks and take risk reduction plan to cope up with these risks.
- **Miscellaneous plans:** This includes making several other plans such as quality assurance plan, configuration management plan, etc.



Crisis meeting of one project group, after 45 minutes discussion the root cause was finally found; nobody understood customer's undocumented API (XML).

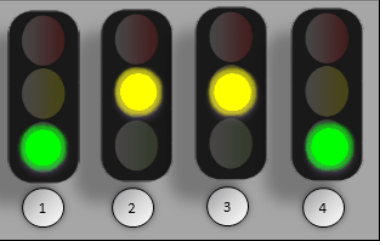
Indicators

TRAFFIC LIGHT KPI DASHBOARD

Products	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	SUM
Product 1	25	174	165	148	95	25	42	69	156	192	129	143	1363
Product 2	99	93	89	45	48	58	89	126	18	85	10	150	910
Product 3	62	87	94	96	94	44	93	55	63	49	154	126	1017
Product 4	100	200	133	85	166	10	142	15	76	69	72	30	1098
Total Sales	286	554	481	374	403	137	366	265	313	395	365	449	4388

Select DEC TRENDS

Product 1
Product 2
Product 3
Product 4

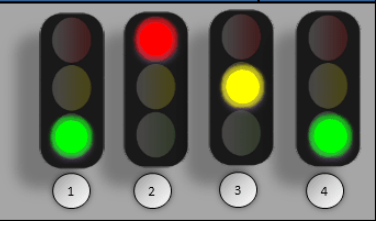


TRAFFIC LIGHT KPI DASHBOARD

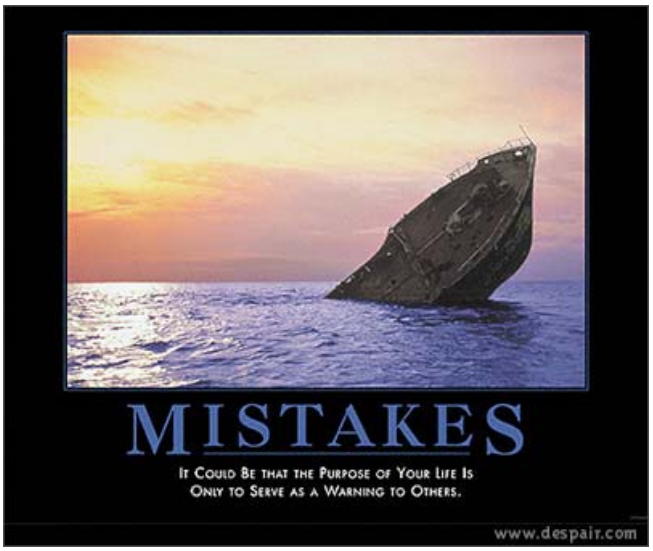
Products	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	SUM
Product 1	25	174	165	148	95	25	42	69	156	192	129	143	632
Product 2	99	93	89	45	48	58	89	126	18	85	10	150	432
Product 3	62	87	94	96	94	44	93	55	63	49	154	126	477
Product 4	100	200	133	85	166	10	142	15	76	69	72	30	694
Total Sales	286	554	481	374	403	137	366	265	313	395	365	449	2235

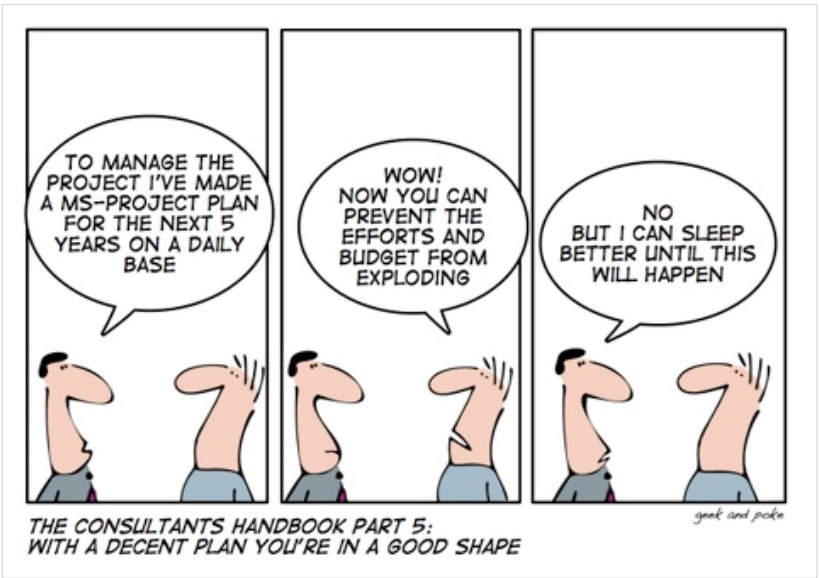
Select JUN TRENDS

Product 1
Product 2
Product 3
Product 4



”Traffic lights” may be used as general indicators of project status. Here based on working hours.





risk management

risk management activities

[Guide to PMBOK, 2017]

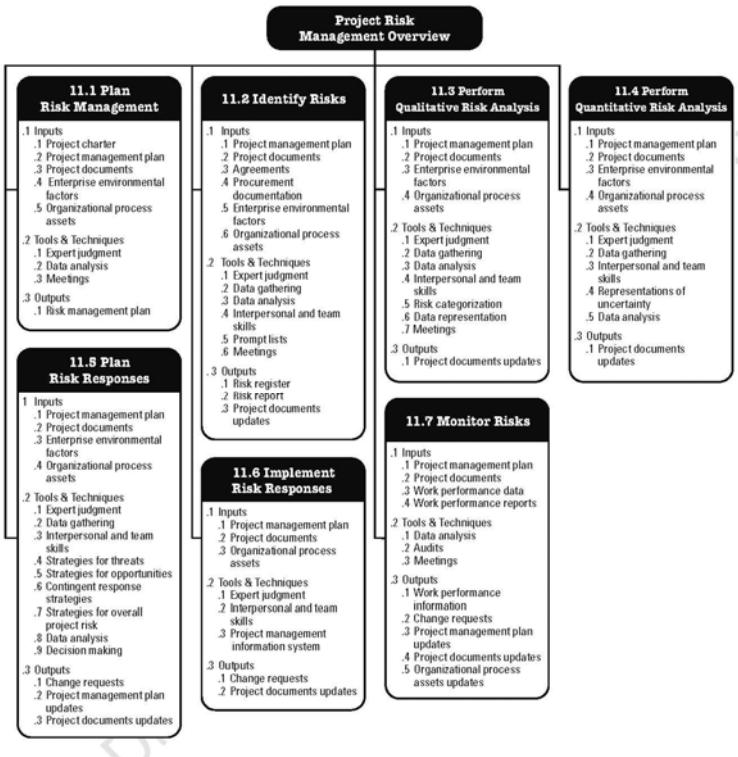


Figure 11-1. Project Risk Management Overview

risk management

Table 11-1. Example of Definitions for Probability and Impacts

SCALE	PROBABILITY	+/− IMPACT ON PROJECT OBJECTIVES		
		TIME	COST	QUALITY
Very High	>70%	>6 months	>$5M	Very significant impact on overall functionality
High	51-70%	3-6 months	$1M-$5M	Significant impact on overall functionality
Medium	31-50%	1-3 months	$501K-$1M	Some impact in key functional areas
Low	11-30%	1-4 weeks	$100K-$500K	Minor impact on overall functionality
Very Low	1-10%	1 week	<$100K	Minor impact on secondary functions
Nil	<1%	No change	No change	No change in functionality

[Guide to PMBOK, 2017]

risk management

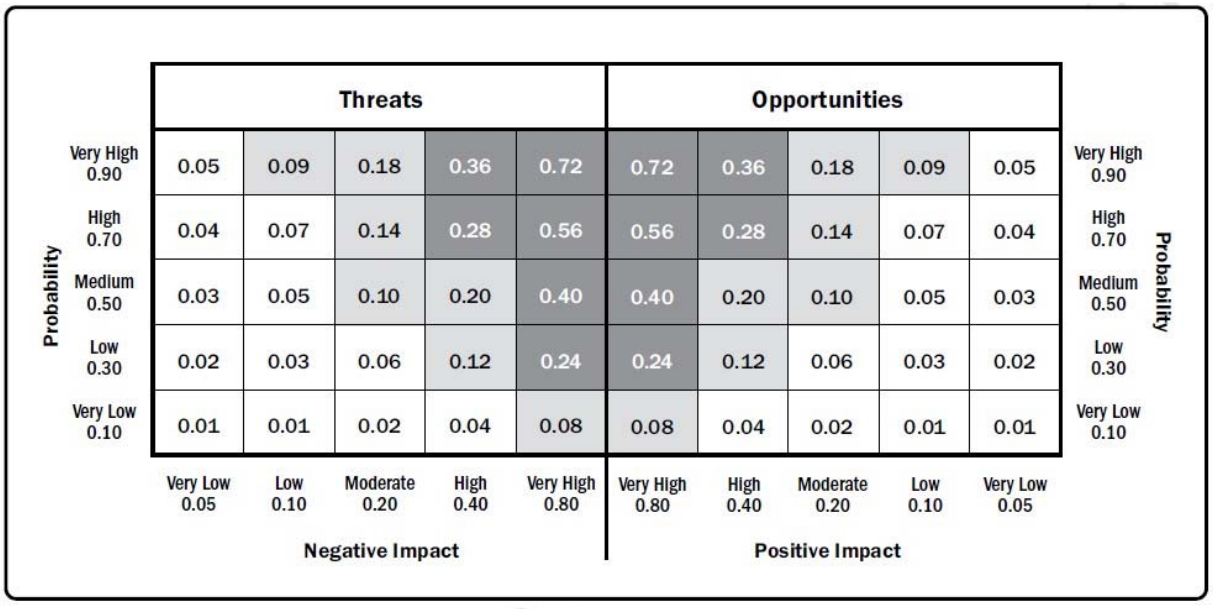


Figure 11-5. Example Probability and Impact Matrix with Scoring Scheme

[Guide to PMBOK, 2017]

Tampereen yliopisto
Tampere University

Size-Complexity Matrix

		COMPLEXITY				
		Very Simple	Simple	Average Complexity	Complex	Very Complex
SIZE	Small	100	250	400	550	625
	Moderate	175	325	475	625	775
	Medium	250	400	550	700	850
	Large	325	475	625	775	925
	Grand	400	550	700	850	1000

The Size-Complexity Matrix provides guidelines for categorizing a project in order to assess the risk and effort. The Size-Complexity Matrix uses a 5-point scale for both size and complexity. The lowest-point project is a simple, small project and has 100 points. The largest and most complex project has 1,000 points. Green means low risk and effort, yellow means medium risk and effort, and red means high risk and effort.

[Standish Group, 2019]

Tampereen yliopisto
Tampere University

Size Guidelines

Size Description		Size
Under $1 million labor	6 or less team members/months	Small
$1 million to $3 million	7 to 12 team members/months	Moderate
$3 million to $6 million	13 to 24 team members/months	Medium
$6 million to $10 million	25 to 50 team members/months	Large
Over $10 Million	Over 50 team members/months	Grand

Guidelines on how to measure the size of a project.

These are the classes for Size-Complexity matrix.

[Standish Group, 2019]

Complexity Guidelines

Environment	Points
Diverse User Base	1
Multiple Team Locations	1
Multiple Stakeholder Locations	1
Uncooperative Peers	2
Uncooperative Stakeholders	3
Scope	**Points**
Many Requirements - Large scope	1
Ambiguous Basic scope	1
Fuzzy Undefined Requirements	1
Diverse and Multifaceted Objectives	2
Breaking New Ground	3

Guidelines on how to measure the complexity of a project.

Risk management

- Risk management is concerned with **identifying risks** and drawing up plans to **minimise their effect** on a project.
- Software risk management is important because of the inherent uncertainties in software development.
  - These uncertainties stem from loosely defined requirements, requirements changes due to changes in customer needs, difficulties in estimating the time and resources required for software development, and differences in individual skills.
- You have to anticipate risks, **understand the impact** of these risks on the project, the product and the business, and take steps to **avoid** these risks.

Risk classification

- There are two dimensions of risk classification
  - The **type** of risk (technical, organizational, ..)
  - what is **affected** by the risk:
- *Project risks* affect schedule or resources;
- *Product risks* affect the quality or performance of the software being developed;
- *Business risks* affect the organisation developing or procuring the software.

Common project risks are derived from
- **group / team** itself (e.g. sickness, motivation, skills,…)
- **customer** (motivation, commitment, clearness of goals,…)
- **technology** (sw, hw, third party components,…)
- **environment** (other parties).

Examples of project, product, and business risks

Risk	Affects	Description
Staff turnover	Project	Experienced staff will leave the project before it is finished.
Management change	Project	There will be a change of organizational management with different priorities.
Hardware unavailability	Project	Hardware that is essential for the project will not be delivered on schedule.
Requirements change	Project and product	There will be a larger number of changes to the requirements than anticipated.
Specification delays	Project and product	Specifications of essential interfaces are not available on schedule.
Size underestimate	Project and product	The size of the system has been underestimated.
CASE tool underperformance	Product	CASE tools, which support the project, do not perform as anticipated.
Technology change	Business	The underlying technology on which the system is built is superseded by new technology.
Product competition	Business	A competitive product is marketed before the system is completed.

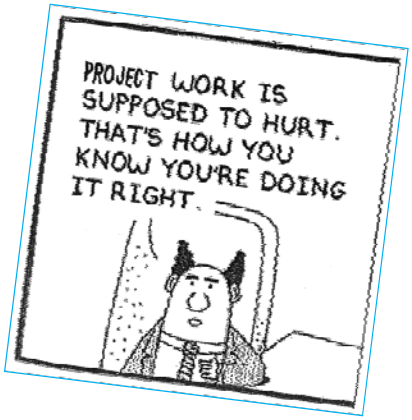
The risk management process

- **Risk identification**
  - Identify project, product and business risks;
- **Risk analysis**
  - Assess the likelihood and consequences of these risks;
- **Risk planning**
  - Draw up plans to avoid or minimise the effects of the risk;
- **Risk monitoring**
  - Monitor the risks throughout the project;

Risk identification

- May be a team activities or based on the individual project manager's **experience**.
- A checklist of common risks may be used to identify risks in a project
  - Technology risks.
  - Organizational risks.
  - People risks.
  - Requirements risks.
  - Estimation risks.

NO PAIN, NO GAIN



Examples of different risk types

Risk type	Possible risks
Estimation	The time required to develop the software is underestimated. (12) The rate of defect repair is underestimated. (13) The size of the software is underestimated. (14)
Organizational	The organization is restructured so that different management are responsible for the project. (6) Organizational financial problems force reductions in the project budget. (7)
People	It is impossible to recruit staff with the skills required. (3) Key staff are ill and unavailable at critical times. (4) Required training for staff is not available. (5)
Requirements	Changes to requirements that require major design rework are proposed. (10) Customers fail to understand the impact of requirements changes. (11)
Technology	The database used in the system cannot process as many transactions per second as expected. (1) Reusable software components contain defects that mean they cannot be reused as planned. (2)
Tools	The code generated by software code generation tools is inefficient. (8) Software tools cannot work together in an integrated way. (9)

Risk analysis

- ✧ Assess probability and seriousness of each risk.
- ✧ Probability may be very low, low, moderate, high or very high.
- ✧ Risk consequences might be catastrophic, serious, tolerable or insignificant.

Usually risks are calculated as

Risk probability * impact = seriousness.

It would be good if you could find some symptom or ”early warning sign” for every risk.

Risk types and examples

Risk	Probability	Effects
Organizational financial problems force reductions in the project budget (7).	Low	Catastrophic
It is impossible to recruit staff with the skills required for the project (3).	High	Catastrophic
Key staff are ill at critical times in the project (4).	Moderate	Serious
Faults in reusable software components have to be repaired before these components are reused. (2).	Moderate	Serious
Changes to requirements that require major design rework are proposed (10).	Moderate	Serious
The organization is restructured so that different management are responsible for the project (6).	High	Serious
The database used in the system cannot process as many transactions per second as expected (1).	Moderate	Serious

Risk types and examples

Risk	Probability	Effects
The time required to develop the software is underestimated (12).	High	Serious
Software tools cannot be integrated (9).	High	Tolerable
Customers fail to understand the impact of requirements changes (11).	Moderate	Tolerable
Required training for staff is not available (5).	Moderate	Tolerable
The rate of defect repair is underestimated (13).	Moderate	Tolerable
The size of the software is underestimated (14).	High	Tolerable
Code generated by code generation tools is inefficient (8).	Moderate	Insignificant

What-if questions

- What if several engineers are **ill** at the same time?
- What if an **economic downturn** leads to budget cuts of 20% for the project?
- What if the performance of **open-source software** is inadequate and the only expert on that open source software leaves?
- What if the company that **supplies** and maintains software components goes out of business?
- What if the **customer fails to deliver** the revised requirements as predicted?

Strategies to help manage risk, 1

Risk	Strategy
Organizational financial problems	Prepare a briefing document for senior management showing how the project is making a very important contribution to the goals of the business and presenting reasons why cuts to the project budget would not be cost-effective.
Recruitment problems	Alert customer to potential difficulties and the possibility of delays; investigate buying-in components.
Staff illness	Reorganize team so that there is more overlap of work and people therefore understand each other's jobs.
Defective components	Replace potentially defective components with bought-in components of known reliability.
Requirements changes	Derive traceability information to assess requirements change impact; maximize information hiding in the design.

Strategies to help manage risk, 2

Risk	Strategy
Organizational restructuring	Prepare a briefing document for senior management showing how the project is making a very important contribution to the goals of the business.
Database performance	Investigate the possibility of buying a higher-performance database.
Underestimated development time	Investigate buying-in components; investigate use of a program generator.

Risk monitoring

- ✧ Assess each identified risks **regularly** to decide whether or not it is becoming less or more probable.
- ✧ Also assess whether the **effects** of the risk have changed.
- ✧ Each key risk should be discussed at management progress meetings.

It is NOT risk management if you write a risk list to Project Plan at the start of a project, and then forget those. **Risk should be considered e.g. at every Sprint start meeting.**

Risk indicators

Risk type	Potential indicators
Estimation	Failure to meet agreed schedule; failure to clear reported defects.
Organizational	Organizational gossip; lack of action by senior management.
People	Poor staff morale; poor relationships amongst team members; high staff turnover.
Requirements	Many requirements change requests; customer complaints.
Technology	Late delivery of hardware or support software; many reported technology problems.
Tools	Reluctance by team members to use tools; complaints about CASE tools; demands for higher-powered workstations.

EN 16601-80:2014
Space project management - Part 80: Risk management

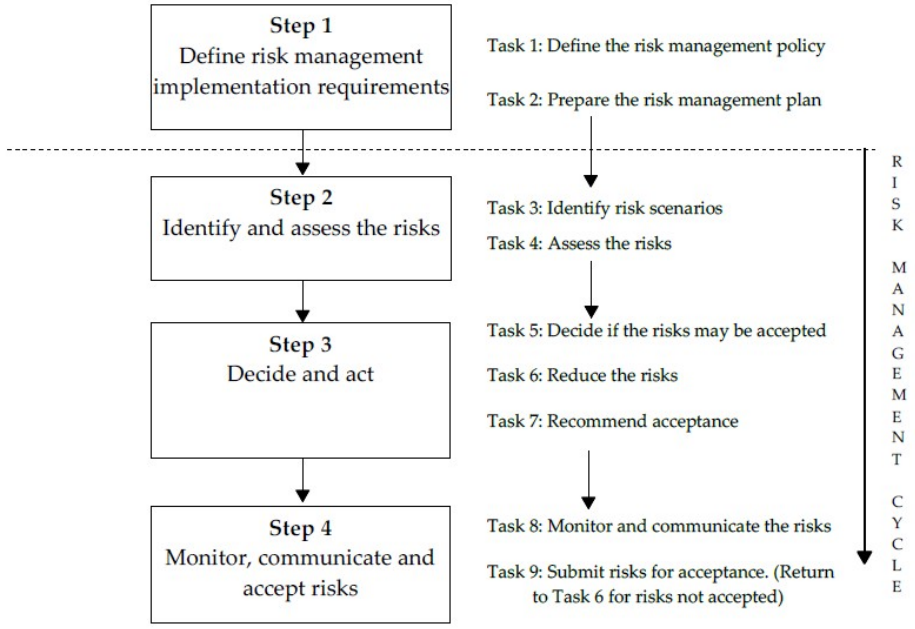
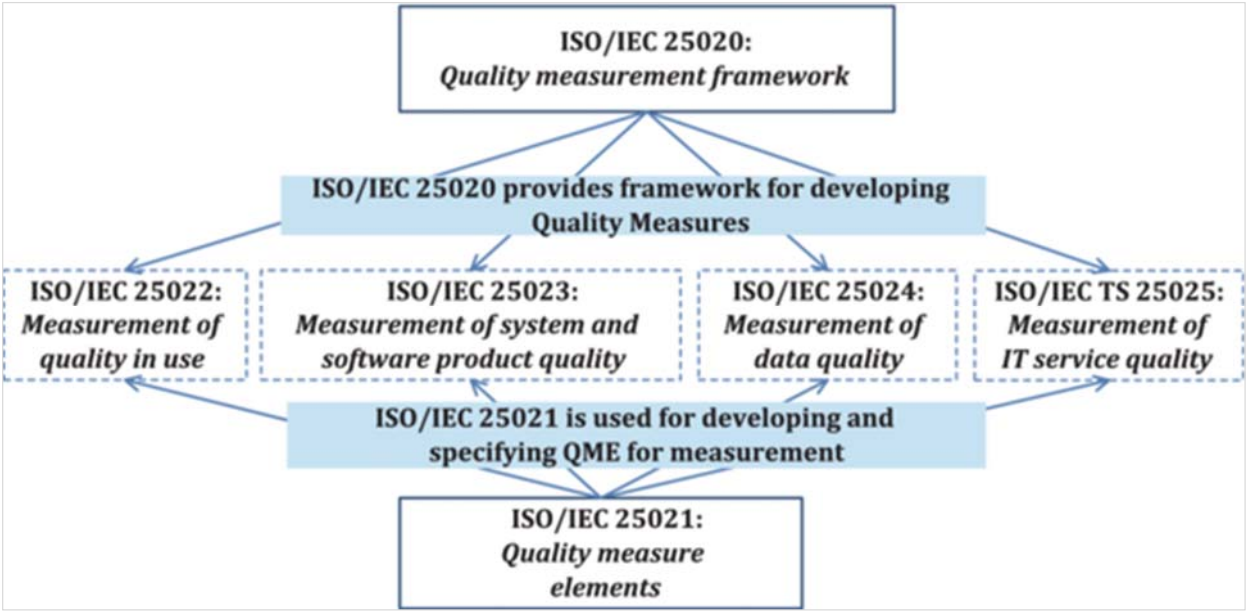


Figure 5-2: The tasks associated with the steps of the risk management process within the risk management cycle

Standards

Quality standards



Notice that standards are like **checklists**, they do not give detailed help to your project.

SUOMEN STANDARDISOIMISLIITTO SFS **STANDARDI** **SFS-ISO 21500**

Suomen Standardisoimisliitto SFS
Finnish Standards Association SFS

Vahvistettu
2012-10-08

1 (1 + 36)

SFS/ICS 03.100

Tämä standardi on vahvistettu englanninkielisenä	*This standard is approved in English*

GUIDANCE ON PROJECT MANAGEMENT

Tämä standardi sisältää kansainvälisen standardin ISO 21500:2012 "Guidance on project management" englanninkielisen tekstin.

Kansainvälinen standardi ISO 21500:2012 on vahvistettu suomalaiseksi kansalliseksi standardiksi.

This standard consists of the English text of the International Standard ISO 21500:2012 "Guidance on project management".

The International Standard ISO 21500:2012 has the status of a Finnish national standard.

Tampereen yliopisto
Tampere University

Some standard…

STANDARDI **SFS-ISO 10006:2018**

Suomen Standardisoimisliitto SFS ry
Finnish Standards Association SFS

Vahvistettu
2018-03-29 2. painos 1 (37)

SFS/ICS 03.100.70; 03.120.10

Korvaa standardin SFS-ISO 10006:en:2004

Replaces the standard SFS-ISO 10006:en:2004

Tämä standardi on vahvistettu englanninkielisenä.

This standard is approved in English.

Quality management. Guidelines for quality management in projects

Tämä standardi sisältää kansainvälisen standardin ISO 10006:2017 "Quality management -- Guidelines for quality management in projects" englanninkielisen tekstin.

This standard consists of the English text of the International Standard ISO 10006:2017 "Quality management -- Guidelines for quality management in projects".

Kansainvälinen standardi ISO 10006:2017 on vahvistettu suomalaiseksi kansalliseksi standardiksi.

The International Standard ISO 10006:2017 has the status of a Finnish national standard.

Tampereen yliopisto
Tampere University

Project management environment

[ISO 21500:2012]

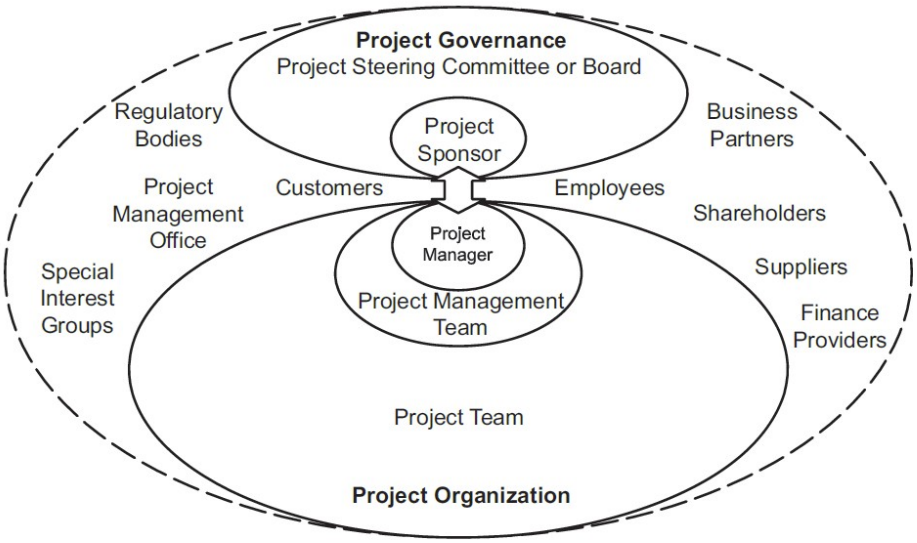


Figure 4 — Project stakeholders

Project management helps business.

[ISO 21500:2012]

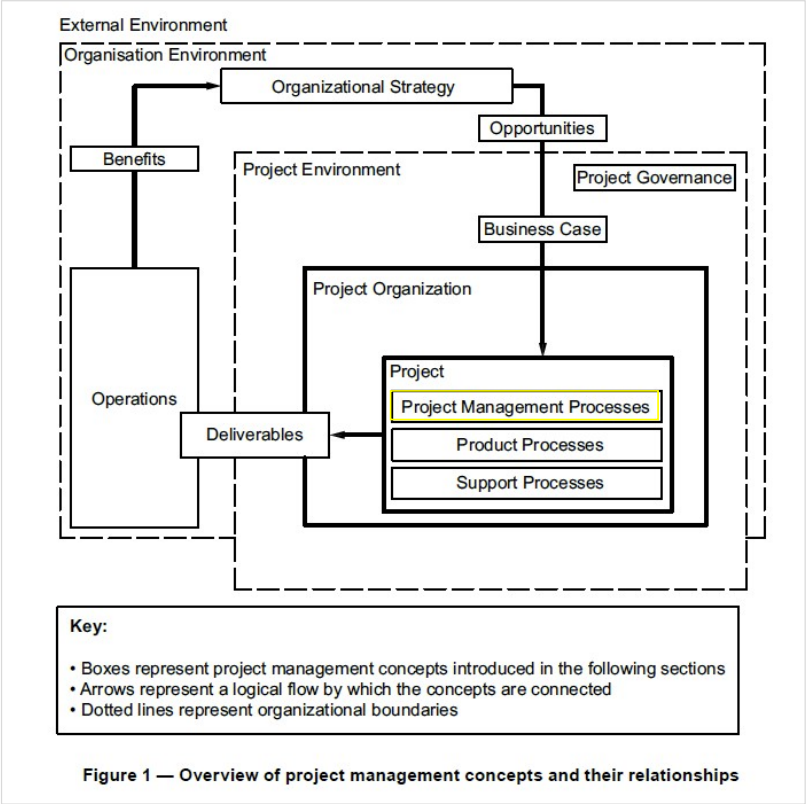


Figure 1 — Overview of project management concepts and their relationships

ISO 1006:2018

STANDARDI

SFS-ISO 10006:2018

Suomen Standardisoimisliitto SFS ry
Finnish Standards Association SFS

Vahvistettu
2018-03-29

2. painos

1 (37)

SFS/ICS 03.100.70; 03.120.10

Korvaa standardin SFS-ISO 10006:en:2004

Replaces the standard SFS-ISO 10006:en:2004

Tämä standardi on vahvistettu englanninkielisenä.

This standard is approved in English.

Quality management. Guidelines for quality management in projects

Tämä standardi sisältää kansainvälisen standardin ISO 10006:2017 "Quality management -- Guidelines for quality management in projects" englanninkielisen tekstin.

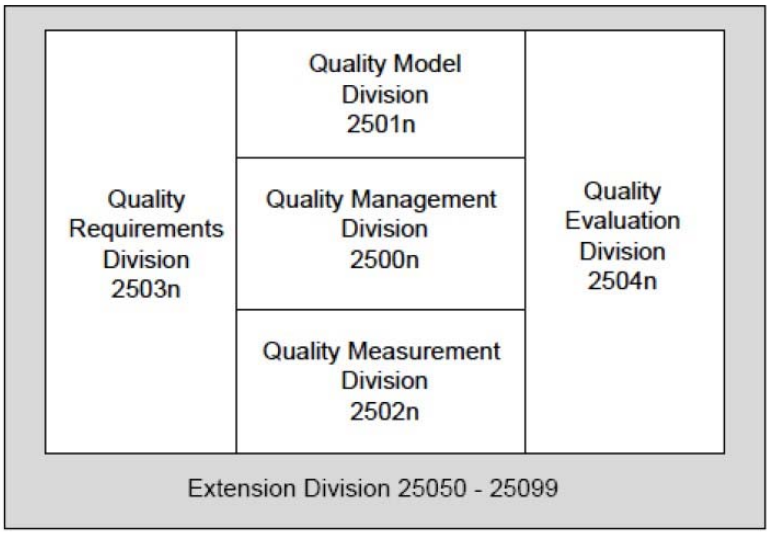
This standard consists of the English text of the International Standard ISO 10006:2017 "Quality management -- Guidelines for quality management in projects".

Kansainvälinen standardi ISO 10006:2017 on vahvistettu suomalaiseksi kansalliseksi standardiksi.

The International Standard ISO 10006:2017 has the status of a Finnish national standard.

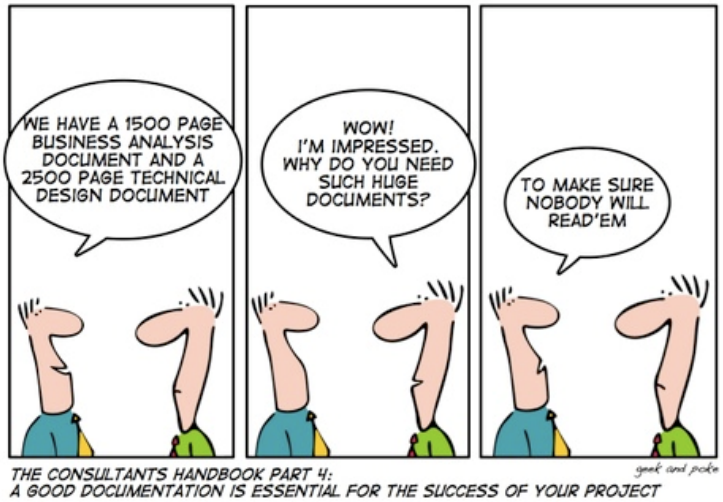
SQuaRE = Software Quality Requirements and Evaluation

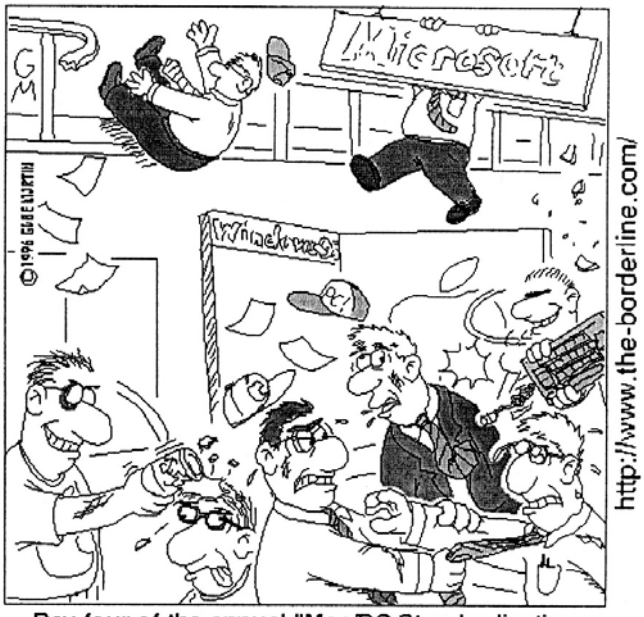
Figure 1 (adapted from ISO/IEC 25000) illustrates the organization of the SQuaRE series representing families of standards, further called divisions.



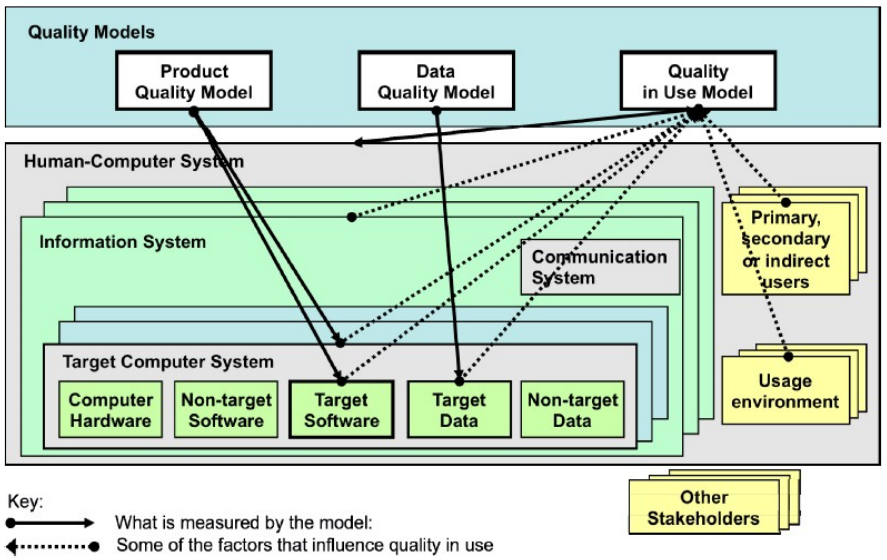
[ISO 25010:2011]

Figure 1 — Organization of SQuaRE series of International Standards





Day four of the annual "Mac/PC Standardization Convention".



NOTE This is conceptually the same as Figure 2 in ISO/IEC 25012 and Figure 5 in ISO/IEC 25030, but a different version that focuses on quality models.

Figure 5 — Targets of quality models

[ISO 25010:2011]

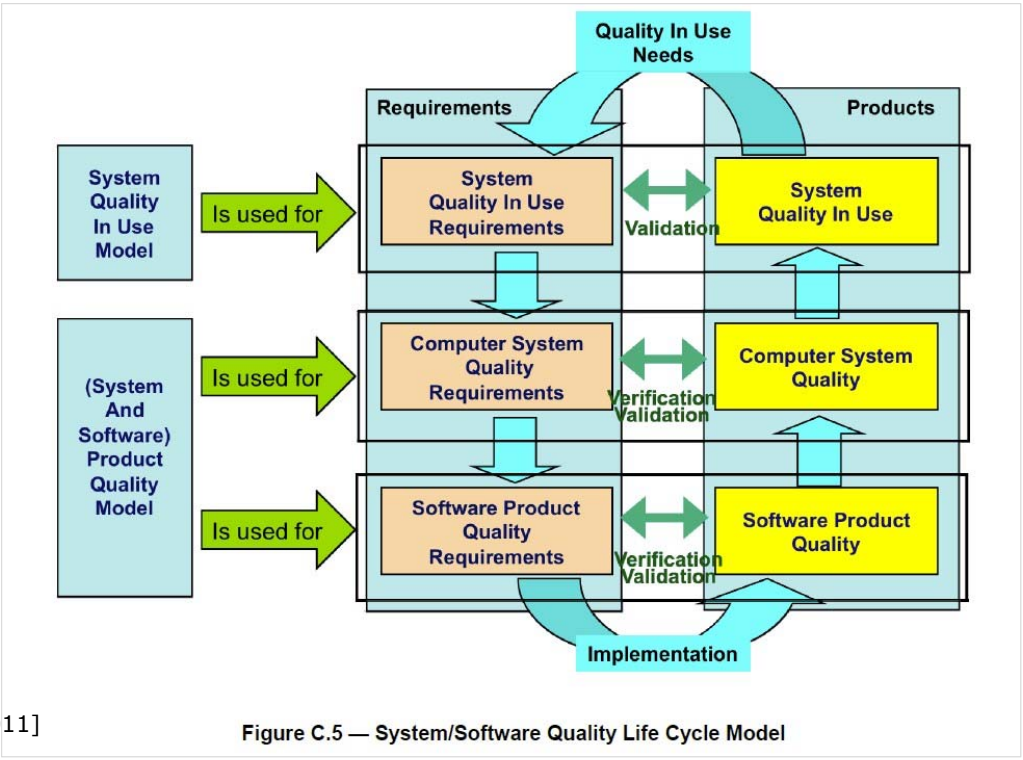


Figure C.5 — System/Software Quality Life Cycle Model

[ISO 25010:2011]

Tampereen yliopisto
Tampere University

ISO/IEC 25010:2011(E)

Product quality

These quality attributes would be considered

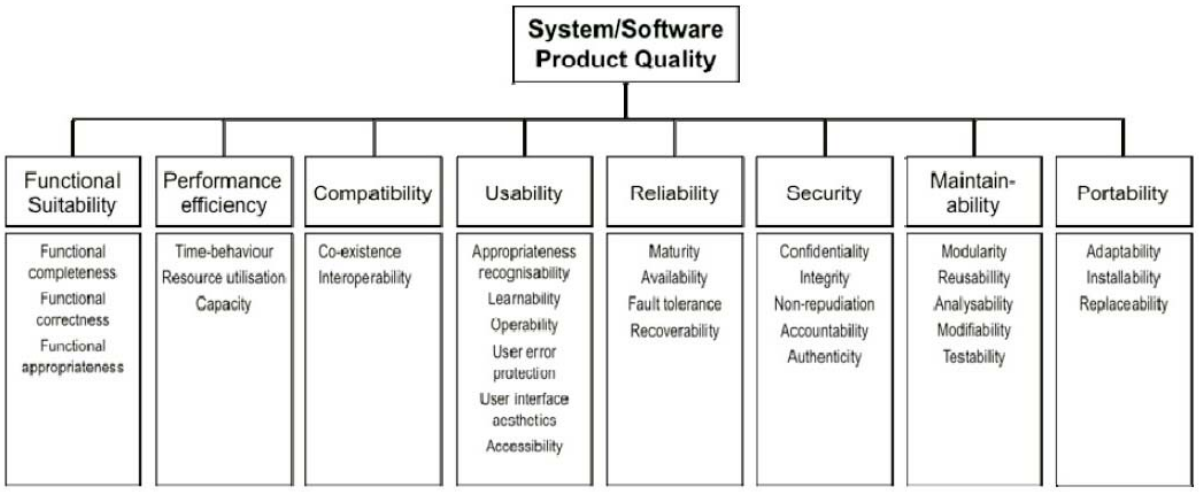


Figure 4 — Product quality model

Tampereen yliopisto
Tampere University

Product quality, in detail 1 / 2

Product Quality - ISO/IEC 25010

Characteristics	Sub-Characteristics	Definition
Functional Suitability	Functional Completeness	degree to which the set of functions covers all the specified tasks and user objectives.
	Functional Correctness	degree to which the functions provides the correct results with the needed degree of precision.
	Functional Appropriateness	degree to which the functions facilitate the accomplishment of specified tasks and objectives.
Performance Efficiency	Time-behavior	degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements.
	Resource Utilization	degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements.
	Capacity	degree to which the maximum limits of the product or system, parameter meet requirements.
Compatibility	Co-existence	degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.
	Interoperability	degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.
Usability	Appropriateness recognisability	degree to which users can recognize whether a product or system is appropriate for their needs.
	Learnability	degree to which a product or system enables the user to learn how to use it with effectiveness, efficiency in emergency situations.
	Operability	degree to which a product or system is easy to operate, control and appropriate to use.
	User error protection	degree to which a product or system protects users against making errors.
	User interface aesthetics	degree to which a user interface enables pleasing and satisfying interaction for the user.
	Accessibility	degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.

Product quality, in detail 2 / 2 [ISO 25010]

Reliability	Maturity	degree to which a system, product or component meets needs for reliability under normal operation.
	Availability	degree to which a product or system is operational and accessible when required for use.
	Fault tolerance	degree to which a system, product or component operates as intended despite the presence of hardware or software faults.
	Recoverability	degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.
Security	Confidentiality	degree to which the prototype ensures that data are accessible only to those authorized to have access.
	Integrity	degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data.
	Non-repudiation	degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later.
	Accountability	degree to which the actions of an entity can be traced uniquely to the entity.
	Authenticity	degree to which the identity of a subject or resource can be proved to be the one claimed.
Maintainability	Modularity	degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.
	Reusability	degree to which an asset can be used in more than one system, or in building other assets.
	Analyzability	degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified.
	Modifiability	degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality.
	Testability	degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met.
Portability	Adaptability	degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments.
	Installability	degree of effectiveness and efficiency in which a product or system can be successfully installed and/or uninstalled in a specified environment.
	Replaceability	degree to which a product can replace another specified software product for the same purpose in the same environment.

Finnish Software Measurement Association

FiSMA

Topic A: Quality measures of software product

- **TOP10 A.1 - Improvement of efficiency of end-user's work**
  - Type: Derived measure
  - Main content: A rate of user tasks, which are supported by the software compared to all other user tasks. A recommended method is a case study.
  - What the measure explains: How well and comprehensively the software fulfils user needs.
- **TOP10 A.2 - End-user satisfaction**
  - Type: Base measure
  - Main content : User experience. Could be divided to sub-measures. A recommended method is Net Promoter.
  - What the measure explains : How successfully the software serves the end-user e.g. usability and accessibility.

FiSMA 2012 8

Finnish Software Measurement Association

FiSMA

Topic B: Software process

- **TOP10 B.1 - Maturity of the software process**
  - Type: Indicator, indirect measure
  - Main content: An operational level derived from a summary of selected processes. Well-known and widely suggested methods are CMMI and SPICE.
  - What the measure explains: Process wise capability of the supplier organisation to deliver products or services.
- **TOP10 B.2 - Agility of the software process**
  - Type: Indicator, indirect measure
  - Main content: A level of agility adaption with the whole software organisation. A recommended method is a survey or an employee inquiry.
  - What the measure explains: Ability to react to external changes or requests.
- **TOP10 B.3 - Improvability of the software process**
  - Type: Indicator, indirect measure
  - Main content: A rate of planned and decided improvement efforts which get completed accordingly. A recommended method is audit.
  - What the measure explains: Capability to execute while there is need to change and develop activities.

FiSMA 2012 9

Finnish Software Measurement Association

FiSMA

Topic C: Management of a software project

- **TOP10 C.1 - Functional size of the software**
  - Type: Derived measure
  - Main content: A size of the software to be developed, acquired, maintained or which is the subject to other activity. A recommended method is FiSMA 1.1 or any other ISO/IEC-standard FSM method (e.g. function points, FP).
  - What the measure explains: Functional size enables comparisons of quality, efficiency and price data of systems of different sizes. Also a value of the software's functionality for the end-user.
- **TOP10 C.2 - Workload of the software project**
  - Type: Base measure
  - Main content: The complete workload of a defined development team in assigned activities during the life cycle of the system. A recommended unit of workload is an hour.
  - What the measure explains: Important source data for schedules, pricing and comparison of productivity.

FiSMA 2012 10

Finnish Software Measurement Association

FiSMA

Topic D: Management of software business

- **TOP10 D.1 - Delivery speed of the software**
  - Type: Indicator, indirect measure
  - Main content: Functional size of the software delivered in the project divided by development time (FP/months).
  - What the measure explains: Delivery speed achieved in the project related to comparable ones; indicates competitiveness of both acquiring and supplying organisations.
- **TOP10 D.2 - Cost efficiency of the software purchase**
  - Type: Indicator, indirect measure
  - Main content: Total cost of the acquired software divided by a functional size, €/FP
  - What the measure explains: The cost efficiency of a project compared to similar ones; indicates competitiveness of both acquiring and supplying organisations.
- **TOP10 D.3 - Efficiency of the development portfolio**
  - Type: Derived measure, partly indicator, indirect measure
  - Main content: Revenues of a development portfolio compared to investments. A recommended method RoI or benefit/cost ratio.
  - What the measure explains: A competence to allocate and address IT efforts in accordance with business goals and value creation.

FiSMA 2012 11

EN 16602-80:2018 Space product assurance. Software product assurance

Just FYI: to use as an ”overkill” checklist.

EN 16602-80:2018 (E)

Software product assurance programme implementation	
5.1 Organization and responsibility	5.5 Procurement
5.2 Software product assurance programme management	5.6 Tools and supporting environment
5.3 Risk management and critical item control	5.7 Assessment and improvement process
5.4 Supplier selection and control	

Software process assurance
6.1 Software development life cycle
6.2 Requirements applicable to all software engineering processes
6.3 Requirements applicable to individual software engineering processes or activities

Software product quality assurance
7.1 Product quality objectives and metrication
7.2 Product quality requirements
7.3 Software intended for reuse
7.4 Standard ground hardware and services for operational system
7.5 Firmware

Figure 4-2: Structure of this Standard

EN 16602-80:2018 (E)

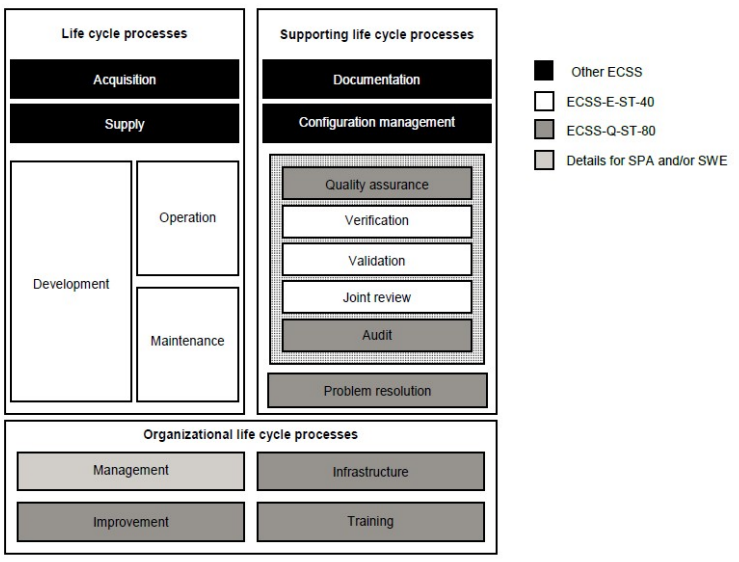
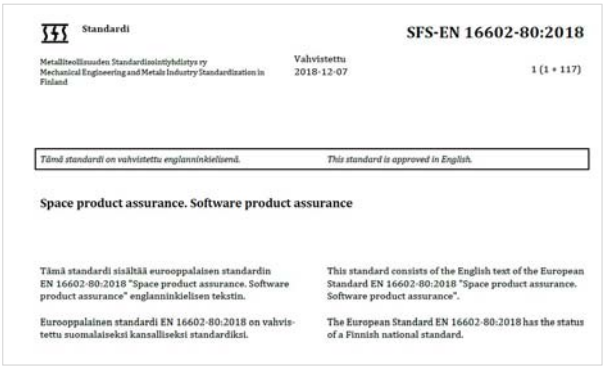


Figure 4-1: Software related processes in ECSS Standards

Tampereen yliopisto
Tampere University

EN-16602-80:2018
Space product assurance. Software product assurance



Standardi

SFS-EN 16602-80:2018

Metalliteollisuuden Standardisointiyhdistys ry
Mechanical Engineering and Metals Industry Standardization in Finland

Vahvistettu
2018-12-07

1 (1 + 117)

Tämä standardi on vahvistettu englanninkielisenä. *This standard is approved in English.*

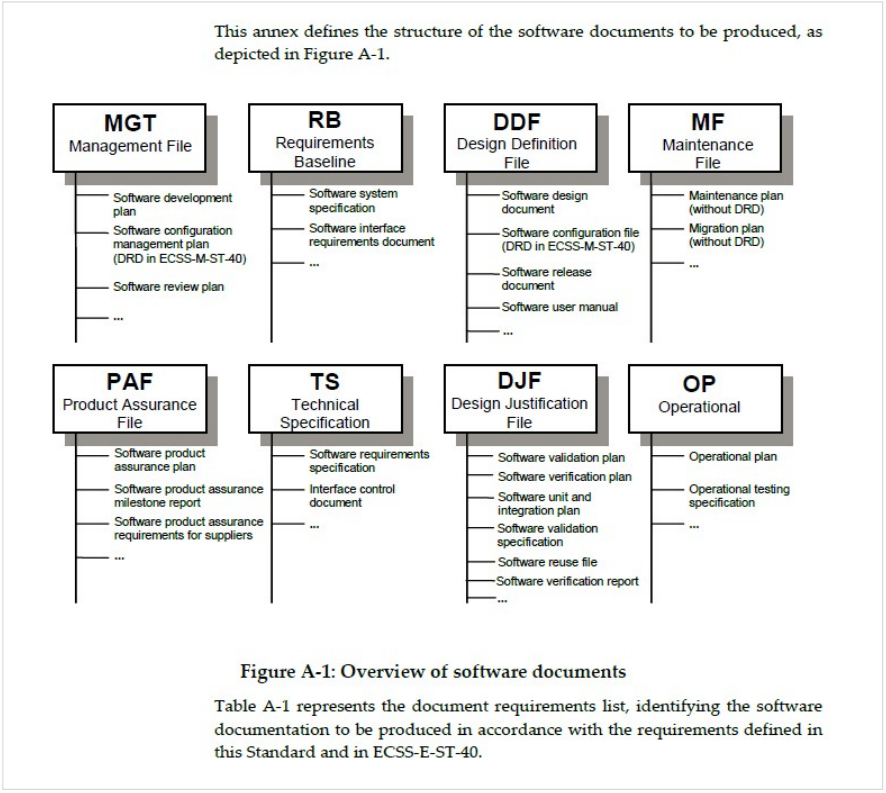
Space product assurance. Software product assurance

Tämä standardi sisältää eurooppalaisen standardin EN 16602-80:2018 "Space product assurance. Software product assurance" englanninkielisen tekstin.

This standard consists of the English text of the European Standard EN 16602-80:2018 "Space product assurance. Software product assurance".

Eurooppalainen standardi EN 16602-80:2018 on vahvistettu suomalaiseksi kansalliseksi standardiksi.

The European Standard EN 16602-80:2018 has the status of a Finnish national standard.



This annex defines the structure of the software documents to be produced, as depicted in Figure A-1.

Figure A-1: Overview of software documents

Table A-1 represents the document requirements list, identifying the software documentation to be produced in accordance with the requirements defined in this Standard and in ECSS-E-ST-40.

Tampereen yliopisto
Tampere University

SFS-EN 16603-40: 2014 Space engineering - Part 40: Software

Processes needed in "heavy duty" software projects.

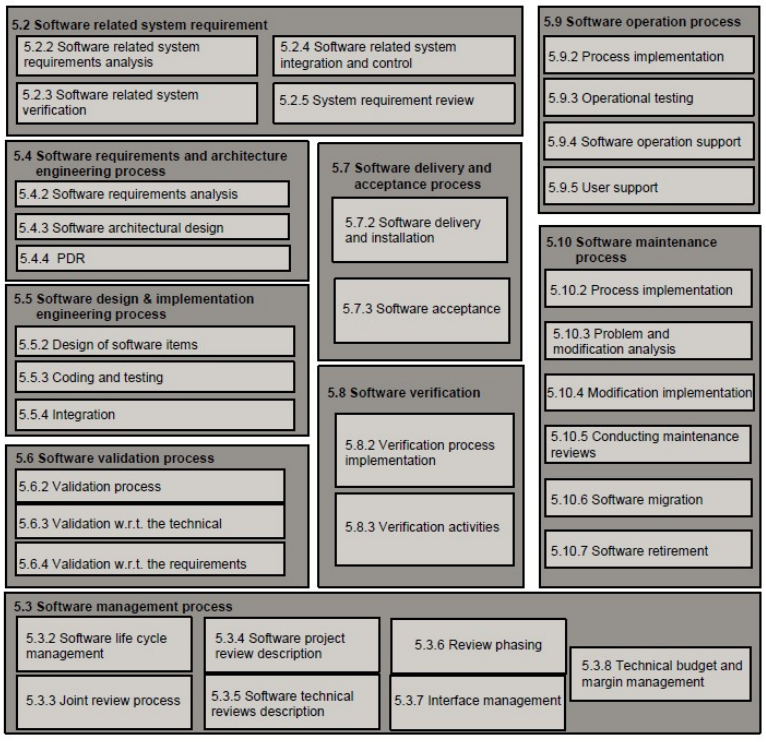


Figure 4-3: Structure of this Standard

Security in standards

There are many standards about ”mission-critical” systems like **Space** systems or **Health Care** systems. You may look at those, for process guidance (start at TUNI electronic Library, **SFS Online**).

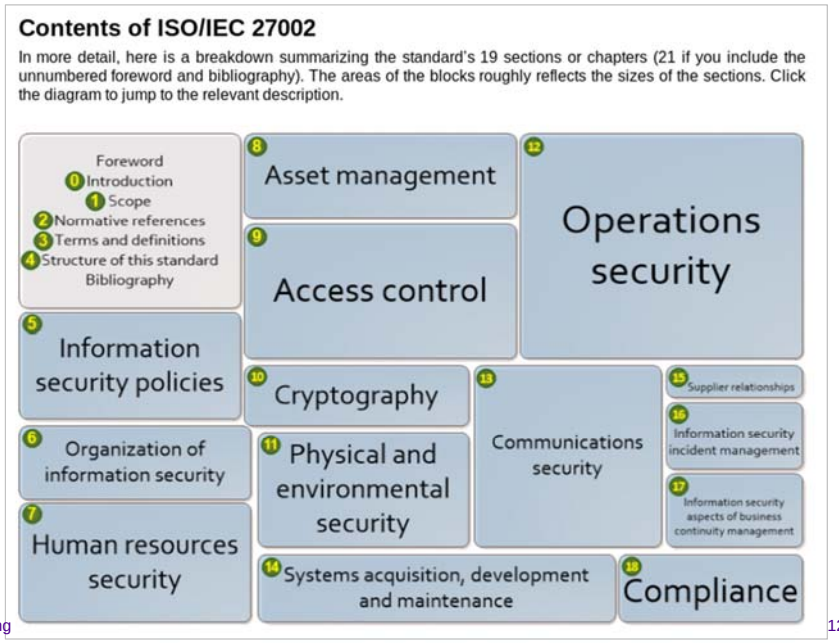
Today both **cybersecurity** and **information security** are also an issue in software development projects, which project manager should also think about (delegate if needed).

One collection www page is

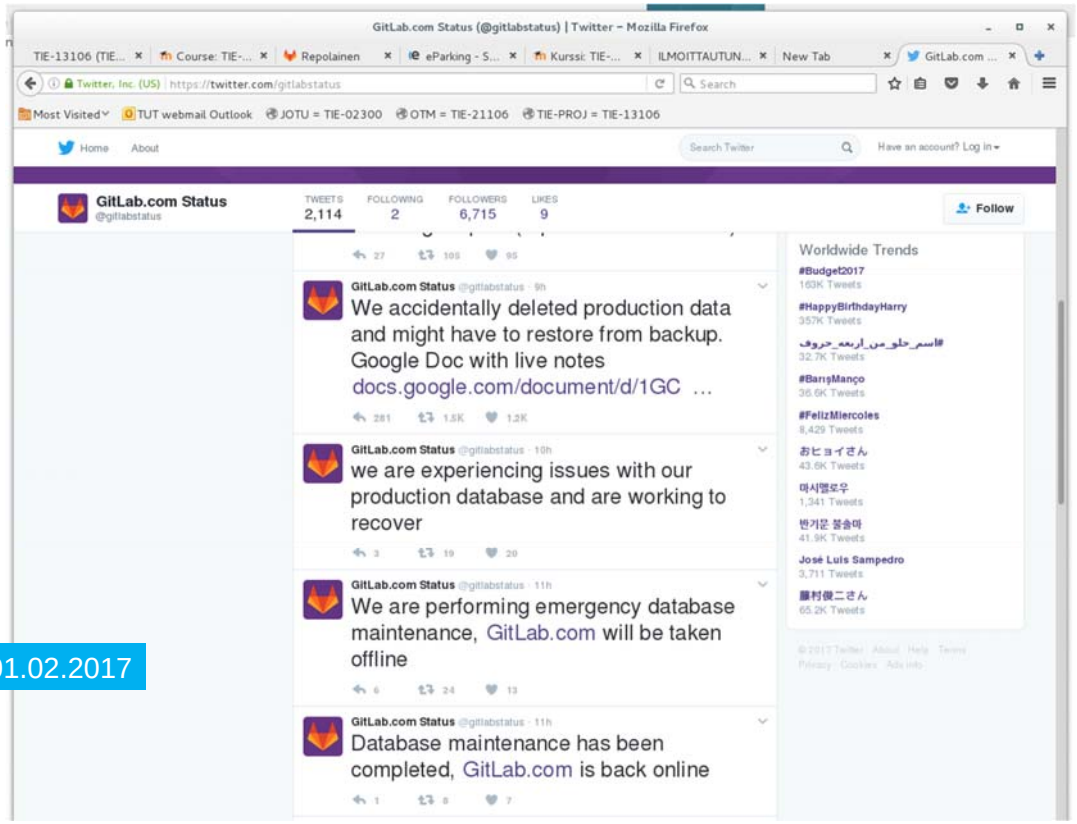
https://www.iso27001security.com/html/iso27000.html

ISO/IEC 27002:2013 — Information technology — Security techniques — **Code of practice for information security controls** *(second edition)*

Althoug these are more important in business, you may consider these also in development projects.



Accidents and ”accidents” may happen even to big companies.



01.02.2017

Tampereen yliopisto
Tampere University

Highlights - What to remember

- project work IS teamwork
- **problems at projects are seldom technical (”hard”), but ”soft”**
- agree project guidelines, regulations and rules, and follow them
- somehow get customer to be committed to the project
- ask help from your teammates, also help them when needed
- remember professional ethics (avoid shortcuts = technical debt)
- be honest, give constructive feedback
- if you see problems in project, tell your superior/boss about them, don’t hide
- understand that sometimes there are ”slow flow” on your or groupmember’s work
- sometimes it is good to use humour in difficult situations (at least in Finland).

”Workload is not killing, it is challenging.”

”Student’s life is not always miserable, sometimes it is just a bad day.”

Tampereen yliopisto
Tampere University

Highlights - What to remember

- think WHAT you need to know or monitor in a project, and measure that wise
- use reasonable metrics (= just enough, not too much
- estimations are estimations, good measurements are exact
- best basis for estimations is experience… which can not be obtained by reading books nor from lectures, you have to work and collect your own data
- estimating size or effort, feel free to ask senior colleagues for their opinion
- discuss in group about size or effort estimations
- you may start estimating of modules/components e.g. ”is this bigger or smaller what I have done earlier”
- coffee room table is an important project management tool ;-)